УНИВЕРСИТЕТ ИТМО

ФГАОУ ВО «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

Факультет программной инженерии и компьютерной техники

ПРИКЛАДНАЯ МАТЕМАТКИА

Лабораторная работа №1

Информационный объем

Лабушев Тимофей

Группа P3302

Санкт-Петербург

2019

Цель работы

Получить практические навыки решения задач на количественное измерение информационного объема текстовой информации.

Задание

1. Реализовать процедуру вычисления энтропии для текстового файла. В процедуре необходимо подсчитывать частоты появления символов (прописные и заглавные буквы не отличаются, знаки препинания рассматриваются как один символ, пробел является самостоятельным символом), которые можно использовать как оценки вероятностей появления символов. Затем вычислить величину энтропии. Точность вычисления – 4 знака после запятой. Обязательно предусмотреть возможность ввода имени файла, для которого будет вычисляться энтропия;
2. Проверить запрограммированную процедуру на нескольких файлах и заполнить таблицу 1 вычисленными значениями энтропии;
3. Вычислить значение энтропии для тех же файлов, но с использованием частот вхождений пар символов и заполнить таблицу 2;
4. Проанализировать полученные результаты.

Исходный код анализа текста

```
#[pyfunction]
fn letter_probabilities(path: &str) -> PyResult<BTreeMap<String, f32>> {
    let text = std::fs::read_to_string(path)?;

    let mut total_num_chars = 0usize;
    let mut frequency_map = HashMap::new();
    for c in iterate_text_chars(&text) {
        total_num_chars += 1;
        *frequency_map.entry(c).or_insert(0) += 1;
    }

    Ok(frequency_map
        .into_iter()
        .map(|(c, freq)| (c.to_string(), freq as f32 / total_num_chars as f32))
        .collect())
}

#[pyfunction]
fn letter_pair_probabilities(path: &str) -> PyResult<BTreeMap<String, f32>> {
    let text = std::fs::read_to_string(path)?;

    let mut total_num_pairs = 0usize;
    let mut frequency_map = HashMap::new();
    for (c1, c2) in iterate_text_chars(&text).tuple_windows() {
        if c1 != ' ' && c1 != '.' && c2 != ' ' && c2 != '.' {
            total_num_pairs += 1;
            *frequency_map.entry(format!("{}{}", c1, c2)).or_insert(0) += 1;
        }
    }

    Ok(frequency_map
        .into_iter()
        .map(|(pair, freq)| (pair, freq as f32 / total_num_pairs as f32))
```

```
        .collect())
}

fn iterate_text_chars<'t>(text: &'t str) -> impl Iterator<Item = char> + 't {
    text.chars().filter_map(|c| match c {
        _ if c.is_ascii_alphabetic() => Some(c.to_ascii_lowercase()),
        _ if c.is_ascii_punctuation() => Some('.'),
        ' ' => Some(' '),
        _ => None,
    })
}
```

Выводы

В ходе работы было установлено, что вероятности встречи символов и пар символов приблизительно равны для трех англоязычных текстов объемом 30, 50, 60 тысяч сиволов, что объясняется их осмысленностью.

Стоит отметить и то, что значение энтропии для пар символов будет мешьше, так как вероятность нахождения пары ниже вероятности нахождения символа.