FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

# Identifikace spamu naivním bayesovským klasifikátorem

Semestrální práce KIV/PC

Štěpán Faragula
A21B0119P

31. prosince 2022

# Obsah

# Kapitola 1

# Zadání

Při volbě zadání semestrální práce jsme měli na výběr z následujících možností:

1. Hledání kořenů rovnice
2. Identifikace spamu naivním bayesovským klasifikátorem
3. Celočíselná kalkulačka s neomezenou přesností

V této práci je popsáno řešení práce **číslo 2**.

## 1.1 Detaily zadání

Naprogramujte v ANSI C přenositelnou **konzolovou aplikaci**, která bude **rozhodovat, zda úsek textu** (textový soubor předaný jako parametr na příkazové řádce) **je nebo není spam**.

Program bude přijímat z příkazové řádky celkem **sedm** parametrů: První dva parametry budou vzor jména a počet trénovacích souborů obsahujících nevyžádané zprávy (tzv. **spam**). Třetí a čtvrtý parametr budou vzor jména a počet trénovacích souborů obsahujících vyžádané zprávy (tzv. **ham**). Pátý a šestý parametr budou vzor jména a počet testovacích souborů. Sedmý para- metr představuje jméno výstupního textového souboru, který bude po dokončení činnosti Vašeho programu obsahovat výsledky klasifikace testovacích souborů.

Program se tedy bude spouštět příkazem

`spamid.exe` ⟨`spam`⟩ ⟨`spam-cnt`⟩ ⟨`ham`⟩ ⟨`ham-cnt`⟩ ⟨`test`⟩ ⟨`test-cnt`⟩ ⟨`out-file`⟩ ↵

Symboly ⟨`spam`⟩, ⟨`ham`⟩ a ⟨`test`⟩ představují vzory jména vstupních souborů. Symboly ⟨`spam-cnt`⟩, ⟨`ham-cnt`⟩ a ⟨`test-cnt`⟩ představují počty vstupních souborů. Vstupní soubory mají následující pojmenování: `vzorN`, kde `N` je celé číslo z intervalu $\langle 1;N \rangle$.

Přípona všech vstupních souborů je `.txt`, přípona není součástí vzoru. Váš program tedy může být během testování spuštěn například takto:

```
spamid.exe spam 10 ham 20 test 50 result.txt ⏎
```

Výsledkem činnosti programu bude textový soubor, který bude obsahovat seznam testovaných souborů a jejich klasifikaci (tedy rozhodnutí, zda je o spam či neškodný obsah – ham).

Pokud nebude na příkazové řádce uvedeno právě sedm argumentů, vypište chybové hlášení a stručný návod k použití programu v angličtině podle běžných zvyklostí (viz např. ukázková semestrální práce na webu předmětu Programování v jazyce C). **Vstupem programu jsou pouze argumenty na příkazové řádce – interakce s uživatelem pomocí klávesnice či myši v průběhu práce programu se neočekává.**

Hotovou práci odevzdejte v jediném archivu typu ZIP prostřednictvím automatického odevzdávacího a validačního systému. Postupujte podle instrukcí uvedených na webu předmětu. Archiv nechť obsahuje všechny zdrojové soubory potřebné k přeložení programu, **makefile** pro Windows i Linux (pro překlad v Linuxu připravte soubor pojmenovaný `makefile` a pro Windows `makefile.win`) a dokumentaci ve formátu PDF vytvořenou v typografickém systému TEX, resp. LATEX. Bude-li některá z částí chybět, kontrolní skript Vaši práci odmítne.

# Kapitola 2

# Analýza úlohy

V úloze máme za úkol **zařadit soubory** do jedné ze dvou tříd – **spam** či **ham**. Je nám výrazně doporučeno použít **naivní bayesovský klasifikátor**.

## 2.1 Definice problému

Testovací soubory budeme klasifikovat podle jejich obsažených slov. Budeme tedy pro každý soubor potřebovat vytvořit jakýsi slovník. Tento slovník bude uchovávat informaci o každém

Nejprve potřebujeme vytvořit slovník Nejprve tedy vytvoříme **slovník klasifikátoru** načtením všech trénovacích souborů. U každého slova budeme uchovávat jeho **počet** a **pravděpodobnost výskytu**.

Testovací soubory budeme třídit podle **slov**, která obsahuje. Z těchto slov vytvoříme **slovník**, který budeme porovnávat se **slovníkem klasifikátoru**. V zadání nám je doporučeno použít **naivní bayesovský klasifikátor**.

K vyřešení úlohy se potřebujeme vyřešit **následující problémy**:

- 
- Volba vhodné struktury pro slovník slov
- Samotná klasifikace testovacích souborů

## 2.2 Zvolení datové struktury slovníku

Slovník musí obsahovat záznam o každém slově, které se načetlo. Potřebujeme využít takovou strukturu, která nám umožní rychle vyhledávat slova v obsáhlém slovníku.

Jelikož ANSI C nenabízí žádnou takovou strukturu, budeme si ji muset naprogramovat sami.

# Kapitola 3

# Popis implementace

Příliš žluťoučký kůň úpěl ďábelské ódy.

# Kapitola 4

# Uživatelská příručka

Příliš žluťoučký kůň úpěl ďábelské ódy.

# Kapitola 5

# Závěr

Příliš žluťoučký kůň úpěl ďábelské ódy.