# Spam filtr

Analýza vypracování úkolu

Autoři: Kateřina Nejedlíková, Matyáš Martan

# Úvod

Zadáním této úlohy bylo vypracovat program, který bude schopný filtrovat e-maily z poskytnutých datasetů na takzvané SPAM e-maily, tedy takové, které obsahují nežádoucí obsah, a HAM e-maily, jejichž obsah je relevantní.

# Princip algoritmu

## Původní záměr

### Bayesovský filtr

Začínali jsme s myšlenkou ve použít pro filtr metodu naivního Bayesova filtrování, zakládajícího se na Bayesově větě. Tato analytická metoda slouží k nalezení pravděpodobnosti jevu na základě jeho dřívějšího výskytu. V případě aplikace na jev, ve kterém se vyskytuje více různých pravděpodobností je ideální, protože bere každý z jevů jako nezávislý, a tedy jako jev, který má stejný vliv na konečný výsledek, jako všechny ostatní. Neexistuje jev, kterému by byla přiznávána větší váha než nějakému jinému jevu.

V případě bayesovského spam filtru je každé jedno slovo „jev“, který má svou vlastní pravděpodobnost, tedy pokud se jedno slovo vyskytne vícekrát, celková váha tohoto slova bude ***n*** násobek původní pravděpodobnosti kdy ***n*** je počet výskytů slova v textu.

Obsah obrázku text, snímek obrazovky, číslo, Písmo

Popis byl vytvořen automatickyKonečné vyhodnocení, zda filtr označí e-mail jako SPAM nebo ne, závisí na celkovém výskytu slov s vyšší pravděpodobností spamu. Pro tento výpočet upravíme základní Bayesův vzorec[[1]](#footnote-1) a spočítáme celkovou pravděpodobnost e-mailu[[2]](#footnote-2). Pokud hodnota této pravděpodobnosti překročí určitou hranici, bude e-mail vyhodnocen jako SPAM. Bayesova pravděpodobnost se dělí na dva typy: apriorní a posteriorní. Oba budou v našem případě použity. Apriorní aplikace pravděpodobnosti bude aplikována poté, co filtr projde „učící“ fází, spočívá ve vycházení z předchozích zkušeností (tedy z učících datasetů). Číslo udávající apriorní pravděpodobnost bude podíl SPAM e-mailů vůči jejich HAM protějškům. Hodnotu apriorní pravděpodobnosti má také každý jeden jev (tedy každé slovo). viz obr. 1

Obrázek

Je rovněž nezbytné určit si seznam běžných slov (tzv. common words), tj. slova, bez kterých jednoduše nelze poskládat jakákoli věta (typicky tedy předložky, spojky a další slova, která se vyskytují ve zprávách nehledě na jejich klasifikaci). Tato slova budou mít, vzhledem k jejich prakticky neustálému výskytu pravděpodobnost SPAM i HAM přibližně 0,5, což v konečném důsledku sníží celkovou SPAM hodnotu e-mailu. Do celkového vzorce pro výpočet je tedy nejvhodnější brát pouze ta slova, která se svou pravděpodobností blíží jednomu, či druhému kraji spektra.

### Aplikace v Pythonu

Vzhledem k tomu, že v u všech e-mailů z testovacího datasetu známe jejich klasifikaci, můžeme vytvořit dva slovníky – jeden pro SPAM a druhý pro HAM. V každém slovníku budou jako klíče uložena všecha slova ze všech e-mailů, jenž jsou klasifikovány jako SPAM, respektive HAM. Jako hodnota bude ke každému klíči uveden počet jeho výskytů ve všech e-mailech z příslušné kategorie.

Dále pak bude vytvořen list pro běžná slova. Jeho inicializace proběhne až poté, co budou kompletně vytvořeny oba předchozí slovníky. Budou do něj přidány ty klíče ze SPAM a HAM slovníků, které se v každém z nich vyskytují dostatečně často, aby je bylo možné označit za běžná slova (důležitý je častý výskyt v obou slovnících). Slova, která budou zařazena do tohoto listu, budou z obou zdrojových slovníků odstraněna.

Následně se z obou slovníků, vytvoří pravděpodobnostní slovníky, kdy pro každé slovo bude vypočítána jeho pravděpodobnost na základě četnosti výskytu a celkového počtu slov v e-mailech.

Pro výpočet apriorní pravděpodobnosti jednoduše spočítáme podíl SPAM, respektive HAM e-mailů v testovacím datasetu vůči celkovému počtu e-mailů.

## Konečné řešení

Po provedení několika testů jsme usoudili, že naše implementace Bayesova filtrování není pro tento úkol ideální, protože filtr nedosahoval uspokojivých výsledků. Z tohoto důvodu jsme ponechali původní princip učení, tak jak byl popsán v předchozí podkapitole, ale upravili jsme způsob vyhodnocení.

\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-

NEJE PART

1. Konečná podoba vzorce pro hodnocení
2. Testovací data, výsledky, úprava parametrů (nová podkapitola?)

\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-

# Rozdělení práce v týmu

## Teoretická příprava

Před zahájením práce na samotném filtru jsme se snažili najít největší možný počet zdrojů pro naši práci. Oba jsme přinesli množství nápadů, z těch následně vybrali ty, které se nám zdály nejefektivnější a nejlépe proveditelné. Pro větší přehlednost Matyáš sepsal pomocí vybraných zdrojů dokument shrnující stručnou strukturu fungování kódu a principu filtru, z něj, a z dalších návodů (na YouTube a na různých internetových stránkách) jsme posléze vycházeli při tvoření programu.

## První fáze filtru

Matyáš vytvořil kostru filtru tak, aby splňovala požadavky na funkčnost a aby filtr disponoval učícím mechanismem, který byl popsán v první kapitole tohoto dokumentu.

## Evaluační funkce a testování

Poté, co filtr procházel testy a byl schopný učení, začala Kateřina pracovat na samotné funkci hodnotící e-maily. Jak již bylo popsáno v předchozí kapitole, rozhodla se nakonec upravit původně zamýšlenou metodu a představila vlastní řešení.

\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-

NEJE PART 2

\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-

# Závěr

Tento projekt byl pro nás oba velmi přínosný, zejména kvůli možnosti aplikace většiny vlastností Pythonu, které jsme v předmětu probírali. V konečném důsledku se nám podařilo vytvořit fungující filtr, jenž byl schopný učení a uspokojivě klasifikoval jednotlivé e-maily.

# Seznam použité literatury

*Bayes' theorem*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-. Dostupné z: <https://en.wikipedia.org/wiki/Bayes%27_theorem>. [cit. 2024-01-04].

*Posterior probability*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-. Dostupné z: <https://en.wikipedia.org/wiki/Posterior_probability>. [cit. 2024-01-04].

*Stop Spam in its Tracks*. Online. In: Medium. 2012-. Dostupné z: <https://medium.com/@dataproducts/stop-spam-in-its-tracks-create-a-naive-bayes-spam-filter-with-python-a5b72b8b4b8f>. [cit. 2024-01-04].

*Laplace smoothing in Naïve Bayes algorithm*. Online. In: JAYASWAL, Vaibhav. Medium. 2012-. Dostupné z: <https://towardsdatascience.com/laplace-smoothing-in-naïve-bayes-algorithm-9c237a8bdece>. [cit. 2024-01-04].

*Naïve Bayes Spam Filter — From Scratch*. Online. In: GARVEY, Mark. Medium. 2012-. Dostupné z: <https://towardsdatascience.com/naïve-bayes-spam-filter-from-scratch-12970ad3dae7>. [cit. 2024-01-04].

Kumar, S., Gupta, K. & Gupta, M. Naïve Bayes Classifier Model for Detecting Spam Mails.*Ann. Data. Sci.* (2023). <https://doi.org/10.1007/s40745-023-00479-z> Dostupné z: <https://link.springer.com/article/10.1007/s40745-023-00479-z#citeas> [cit. 2024-01-04].

1. 

   Pr(S|W) je pravděpodobnost, že e-mail je spam, pokud obsahuje dané slovo (W). Pr(H) představuje pravděpodobnost, že zpráva není spam, Pr(H|W) je pravděpodobnost, se kterou se dané slovo vyskytuje v e-mailech, které nejsou spam. [↑](#footnote-ref-1)
2. 

   Kdy v tomto vzorci p je celková pravděpodobnost, že zpráva je spam a (p1, …, pn) jsou dílčí pravděpodobnosti pro jednotlivá slova v textu. Kvůli reprezentaci extrémně malých čísel v CPU se vyplatí použít spíše upravený vzorec: kdy Obsah obrázku černá, tma

   Popis byl vytvořen automaticky [↑](#footnote-ref-2)