Homework 2
Corpus Analysis

Swesan Pathmanathan
pathmans@mcmaster.ca

1 Dataset

For this assignment, I analyzed a Linux system log dataset categorized into error and normal log entries. The dataset was derived from the Linux.txt file used in Homework 1, which contains authentication and kernel logs from a Linux system.

1.1 Dataset Source and Collection

The original dataset is a Linux system log file containing 308,039 raw tokens collected from authentication and kernel logs. To create distinct categories for corpus analysis, I split the log file into two categories based on the presence of error-related keywords:

- **Error logs**: Log entries containing keywords such as "error", "failed", "failure", "killed", "denied", "refused", "timeout", "exception", "critical", or "alert"
- **Normal logs**: Log entries that do not contain error keywords, representing successful operations, system startup messages, and routine kernel operations

This categorization approach is meaningful because it distinguishes between system events that indicate problems (errors) and those that represent normal system operation. This distinction is valuable for understanding how different types of system events are expressed in log language.

1.2 Document Splitting

Each log entry (line) in the original file was treated as a separate document. This approach is appropriate for system logs because each line typically represents a complete, self-contained event. The splitting was performed by:

1. Reading the log file line by line
2. Classifying each line as error or normal based on keyword presence
3. Writing each line to a separate document file in the appropriate category directory
4. Limiting each category to 500 documents to ensure balanced representation and manageable file sizes

This resulted in 500 documents per category, meeting the requirement of at least 100 documents per category.

1.3 Dataset Statistics

Table 1: Dataset statistics by category

Category	Documents	Avg Tokens/Doc	Total Words
Error	500	13.1	6543
Normal	500	10.5	5247

The dataset contains 1000 total documents (500 per category) with an average of 13.1 tokens per document for error logs and 10.5 tokens per document for normal logs. The error category has a slightly higher average token count, which may reflect that error messages tend to be more verbose, containing additional diagnostic information.

2 Methodology

2.1 Preprocessing

The preprocessing pipeline reuses and extends functionality from Homework 1. The following normalization options were available:

- **Lowercasing**: Converts all tokens to lowercase to reduce vocabulary size.
- **Stopword Removal**: Removes common English stopwords using the NLTK stopword list.
- **Stemming**: Applies Porter stemming to reduce words to their root forms.
- **Lemmatization**: Applies WordNet lemmatization to normalize words to their dictionary form (mutually exclusive with stemming).
- **Digit Removal**: Removes tokens that consist only of digits (custom option from HW-1).

Tokenization is performed using manual whitespace splitting, as in Homework 1. All file operations use UTF-8 encoding with error replacement for robustness.

2.2 Bag-of-Words Conversion

Documents were converted to bag-of-words format using scipy sparse matrices for memory efficiency. Three representation types were supported:

- **Count**: Raw token counts per document
- **Binary**: Binary indicators (1 if token appears, 0 otherwise)
- **TF-IDF**: Term frequency-inverse document frequency transformation

2.3 Naive Bayes Analysis

Naive Bayes probabilities were computed for each word and category:

- $P(w|c)$: Probability of word w given category c (with add-one smoothing)
- $P(w|C_o)$: Probability of word w given other categories (equation 2 from assignment)
- Log-likelihood ratio: $\text{LLR}(w, c) = \log(P(w|c)) - \log(P(w|C_o))$

The top 10 words per category were extracted based on LLR scores.

2.4 Topic Modeling

Latent Dirichlet Allocation (LDA) was performed using the `gensim` library. The number of topics was chosen based on coherence optimization or manual inspection. For each topic:

- Top 25 terms with probabilities were extracted
- Topic distributions per document were computed
- Average topic distributions per category were calculated
- Top 3-5 topics per category were identified

Visualizations were generated using `pyLDAvis`.

2.5 Libraries Used

- `gensim`: LDA topic modeling
- `pyLDAvis`: Topic visualization
- `scipy`: Sparse matrix operations
- `scikit-learn`: TF-IDF transformation
- `nltk`: Preprocessing (stopwords, stemming, lemmatization)
- `numpy`, `pandas`: Data manipulation

3 Results and Analysis

3.1 Naive Bayes Analysis

The Naive Bayes log-likelihood ratio analysis identified words that are most strongly associated with each category. Table 2 shows the top 10 words per category sorted by their LLR scores.

Table 2: Top 10 words per category by log-likelihood ratio

Category	Rank	Word (LLR)
Error	1	`authentication` (5.9178)
Error	2	`uid=0` (5.8100)
Error	3	`euid=0` (5.8100)
Error	4	`failure;` (5.8100)
Error	5	`ruser=` (5.8100)
Error	6	`logname=` (5.8100)
Error	7	`tty=nodevssh` (5.8100)
Error	8	`user=root` (5.2068)
Error	9	`jul` (4.2153)
Error	10	`failed` (3.7789)
Normal	1	`succeeded` (4.2168)
Normal	2	`device` (4.1987)
Normal	3	`node` (4.0626)
Normal	4	`11:31:47` (3.9756)
Normal	5	`startup` (3.8803)
Normal	6	`kernel:` (3.7340)
Normal	7	`named[2275]:` (3.6254)
Normal	8	`creating` (3.5926)
Normal	9	`driver` (3.5926)
Normal	10	`06:06:24` (3.5587)

Main takeaways: The error category is strongly characterized by authentication-related terms ("authentication", "uid=0", "euid=0", "failure;", "ruser=", "logname=", "tty=nodevssh") and failure indicators ("failed"). These terms reflect SSH authentication failures and permission issues. The normal category is characterized by successful operations ("succeeded", "startup", "creating") and system components ("device", "node", "kernel:", "driver", "named[2275]:"), indicating routine system operations and service initialization.

3.2 Topic Modeling

Latent Dirichlet Allocation with 10 topics was applied to the corpus. Table 3 shows the top 15 terms for each topic, with manually assigned topic labels based on the most prominent terms.

Table 3: Top terms per topic (showing first 15 terms)

Topic	Rank	Term (Probability)
System Startup	1	kernel: (0.0760)
	2	9 (0.0609)
	3	10 (0.0368)
	4	06:06:20 (0.0185)
	5	succeeded (0.0184)
	6	11:31:44 (0.0180)
	7	startup (0.0170)
	8	11:31:46 (0.0153)
	9	bios (0.0138)
	10	hda: (0.0128)
	11	lifo (0.0127)
	12	zone: (0.0127)
	13	pages, (0.0127)
	14	0 (0.0127)
	15	06:06:24 (0.0119)
Kernel Operations	1	kernel: (0.0500)
	2	9 (0.0381)
	3	10 (0.0256)
	4	rhost=61.153.202.254 (0.0185)
	5	10:18:09 (0.0185)
	6	version (0.0183)
	7	1 (0.0143)
	8	irq (0.0134)
	9	linux (0.0123)
	10	pci (0.0112)
	11	user=root (0.0112)
	12	rhost=60.30.224.116 (0.0106)
	13	authentication (0.0099)
	14	ruser= (0.0099)
	15	failure; (0.0099)
Device Detection	1	kernel: (0.0568)
	2	10 (0.0464)
	3	9 (0.0351)
	4	11:31:45 (0.0187)
	5	06:06:21 (0.0185)
	6	found (0.0163)
	7	check (0.0159)
	8	machine (0.0159)
	9	enabled (0.0159)
	10	11:31:47 (0.0151)
	11	intel (0.0149)
	12	/dev/hda, (0.0144)
	13	device: (0.0144)
	14	network: (0.0143)
	15	10:56:41 (0.0123)
Process Management	1	9 (0.0836)
	2	device (0.0573)
	3	node (0.0496)
	4	alert (0.0332)

Table 4 shows the top 5 topics for each category, representing the average topic distribution across documents in that category.

Table 4: Top 5 topics per category

Category	Rank	Topic (Probability)
Error	1	Topic 6 (0.2956)
Error	2	Topic 4 (0.2851)
Error	3	Topic 9 (0.1878)
Error	4	Topic 8 (0.1055)
Error	5	Topic 3 (0.0624)
Normal	1	Topic 5 (0.3466)
Normal	2	Topic 7 (0.1728)
Normal	3	Topic 3 (0.1711)
Normal	4	Topic 0 (0.0946)
Normal	5	Topic 1 (0.0702)

Main takeaways: The error category is dominated by authentication-related topics (Topics 6, 4, and 9: SSH Authentication, Authentication Failures, and Authentication Errors), which together account for 76.8% of the topic distribution. The normal category is dominated by Topic 5 (Network Services) at 34.7%, followed by Topic 7 (Kernel Messages) and Topic 3 (Process Management), reflecting routine system operations. The topic modeling successfully separated authentication failures from successful system operations, demonstrating that LDA can identify meaningful thematic patterns in log data.

An interactive visualization of the topic model is available in `output/topic`$_{m}odeling/lda\_visualization.html, whicha$

3.3 Experimentation

To understand the effects of different preprocessing configurations, I tested five normalization approaches with topic modeling, measuring coherence scores as an indicator of topic quality. Table 5 shows the results.

Table 5: Comparison of preprocessing configurations

Configuration	Vocab Size	Coherence
raw	1009	0.4319
lowercase	970	0.4281
lowercase_stopwords	934	0.4559
lowercase_stopwords_stem	913	0.3798
lowercase_stopwords_lemmatize	929	0.4701

Main takeaways: The configuration with lowercase, stopword removal, and lemmatization achieved the highest coherence score (0.4701), indicating that this preprocessing combination produces the most interpretable topics. Lowercasing and stopword removal reduced vocabulary size from 1009 to 934 tokens while improving coherence from 0.4319 to 0.4559. Interestingly, lemmatization (coherence 0.4701) outperformed stemming (coherence 0.3798), suggesting that preserving valid dictionary forms is more beneficial for log text than aggressive stemming. The raw configuration had the largest vocabulary

but lower coherence, indicating that preprocessing improves topic quality by reducing noise and normalizing word forms.

4 Discussion

4.1 Findings About the Dataset

The analysis revealed clear linguistic differences between error and normal log entries, demonstrating that system logs contain distinct "languages" for different types of events.

What makes error logs unique: Error logs are dominated by authentication-related language. The most distinctive words in error logs are terms like "authentication", "uid=0", "euid=0", "failure;", "ruser=", "logname=", and "tty=nodevssh". These terms form a consistent pattern describing failed SSH login attempts, where the system records user information (user ID, effective user ID, remote user, login name) along with the failure indicator. This creates a "failure language" that is highly structured and repetitive, making it easy to identify error conditions even without understanding the technical details.

What makes normal logs unique: Normal logs use language focused on successful operations and system components. Words like "succeeded", "startup", "creating", "device", "node", "kernel:", and "driver" indicate routine system activities: services starting successfully, hardware being detected, and system components being initialized. The presence of timestamps (like "11:31:47" and "06:06:24") in the top words suggests that normal logs often contain temporal markers, possibly because successful operations are logged with more precise timing information.

How they differ: The most striking difference is that error logs are descriptive (they describe what went wrong and who was involved), while normal logs are action-oriented (they describe what the system is doing). Error logs read like incident reports ("authentication failure for user=root from remote host X"), while normal logs read like activity logs ("device found, driver loaded, service started").

Topics that emerged: The topic modeling discovered 10 distinct themes in the logs. Error logs are primarily composed of three authentication-related topics (SSH authentication failures, authentication errors, and Kerberos authentication issues), which together account for about 77% of error log content. Normal logs are primarily composed of network services (35%), kernel messages (17%), and process management (17%), reflecting routine system operations. This separation is so clear that you could almost classify a log entry as error or normal just by looking at which topics it belongs to.

Interesting patterns: One interesting finding is that error logs tend to be slightly longer (13.1 tokens vs 10.5 tokens on average), suggesting that errors require more explanation. Also, the topic model successfully separated different types of authentication failures (SSH vs Kerberos), showing that even within the error category, there are distinct sub-patterns. The presence of specific IP addresses and hostnames in topic terms suggests that certain error patterns are associated with particular sources, which could be useful for security analysis.

4.2 Personal Lessons Learned

Data collection and preprocessing: I learned that creating meaningful categories from raw log data requires careful consideration of what makes categories distinct. Simply splitting by keywords worked well for this dataset, but I realized that for more complex logs, more sophisticated categorization (e.g., by log level, component, or time period) might be necessary. The preprocessing choices significantly impacted results: removing stopwords and applying lemmatization improved topic coherence, while stemming actually hurt performance, likely because log text contains many technical terms and proper nouns that don't benefit from aggressive stemming.

Understanding the results: Interpreting Naive Bayes LLR scores was initially challenging. High LLR scores indicate words that are much more likely in one category than the other, but I had to learn that very high scores (like 5.8) don't necessarily mean the word is "better"—they just mean it's more distinctive. For topic modeling, I discovered that manually assigning topic labels based on top terms was crucial for understanding what each topic represents. The raw topic numbers (Topic 0, Topic 1, etc.) are meaningless without interpretation.

Technical insights: I gained appreciation for sparse matrix representations—the document-term matrix was 99.36% sparse, meaning most entries were zero. Using SciPy sparse matrices was essential for memory efficiency. I also learned that topic coherence is a useful metric for comparing preprocessing configurations, but it's not perfect—sometimes topics with lower coherence scores are still interpretable. The experimentation showed that there's no one-size-fits-all preprocessing pipeline; the best configuration depends on the specific corpus and analysis goals.

Limitations encountered: One limitation was that the dataset, while meeting the 100-document requirement, is relatively small (1000 documents total). This may have affected topic quality, as LDA typically benefits from larger corpora. Also, the simple keyword-based categorization might have misclassified some entries—a log line containing "error" in a different context (e.g., "no error detected") would be incorrectly labeled as an error. For production use, more sophisticated classification would be needed.

What was new and interesting: This was my first experience with topic modeling on real-world data. I was surprised by how well LDA separated authentication failures from normal operations—the topic distributions were so distinct that they could serve as features for automatic log classification. The visualization tools (pyLDAvis) helped me understand topic relationships, though I encountered some Python 3 compatibility issues that required workarounds. Overall, the assignment demonstrated that NLP techniques can extract meaningful patterns from what initially appears to be unstructured log text.

Generative AI Usage Disclosure

Generative AI tools (specifically Cursor AI Assistant) were used extensively throughout this assignment for conceptual clarification, debugging guidance, code structure assistance, and documentation wording. All final code, decisions, and interpretations were written and verified by the student.

AI Usage Details

- **Model**: Cursor AI Assistant (Claude-based model)

- **Provider**: Anthropic (via Cursor IDE)
- **Primary uses**:
  - Code structure and architecture design for corpus analysis pipeline
  - Debugging Python import errors and dependency issues
  - Assistance with LaTeX table generation and formatting
  - Clarification of Naive Bayes LLR calculations and topic modeling concepts
  - Documentation and README writing assistance
- **Time used**: Approximately 4–6 hours of active AI assistance over the course of assignment completion
- **Estimated emissions**: Based on typical usage patterns, approximately 0.5–1.0 kg CO2 equivalent (rough estimate based on model inference costs)

Student Contribution All analysis results, interpretations, and conclusions presented in this report are my own work. The AI assistant was used as a tool for implementation and clarification, but all analytical insights, discussion points, and final code decisions were made and verified by me.