Project Report: AI-Powered Customer Support for HelpNow

Prepared for: VP of Operations, HelpNow Leadership **Date:** August 25, 2025 **Project Team:** Data Science Interns

1. Executive Summary

The objective of this project was to explore the use of machine learning to scale HelpNow's customer support operations. After a comprehensive analysis of 8,469 support tickets, we have developed a suite of AI-powered tools ready for deployment and have formulated a strategic recommendation based on a deep understanding of the company's data.

Key Findings & Recommendations:

- **Ticket Categorization:** We have built a **highly effective hybrid classification system**. A "Gatekeeper" AI model can instantly and with near-perfect accuracy (**100%**) identify and route all **"Account Access"** tickets. However, we have conclusively proven that the textual content of the remaining tickets (`Billing`, `General Query`, `Technical Issue`) is not distinct enough for reliable automated classification.
    - **Recommendation:** Immediately deploy the "Gatekeeper" model. Tickets not identified as "Account Access" should be routed to a single human **Triage Queue** for fast, final categorization. This leverages AI for the easy cases and human intelligence for the nuanced ones, maximizing both efficiency and accuracy.
- **Duplicate Ticket Detection:** We have developed a powerful semantic system that can identify potential duplicate tickets with very high confidence (**>99% similarity score** for near-identical issues).
    - **Recommendation:** Integrate this system to flag potential duplicates for support agents, allowing them to merge tickets and reduce redundant work.
- **Semantic Search for Resolutions:** We have built a proof-of-concept semantic search engine that allows agents to find the most relevant past solutions for new tickets.
    - **Recommendation:** Deploy this as a tool for new and existing support agents to accelerate problem-solving and improve training.
- **Model Choice:** Across all tasks, modern **Transformer-based models (DistilBERT)** vastly outperformed traditional methods (TF-IDF, Naive Bayes). The initial higher cost of computation is justified by the significant increase in semantic understanding and overall system performance.

2. Project Deep Dive: The Journey & Key Insights

A. The Ticket Categorization Challenge

We began by testing a range of models, from simple baselines to complex neural networks, to automatically classify tickets into four categories.

- **Initial Failure:** All initial models, including Logistic Regression, Naive Bayes, and a standard LSTM, failed to perform adequately. They were heavily biased by the majority "Billing" class and could not distinguish the minority classes.
- **The Critical Insight:** Our investigation culminated in testing a state-of-the-art DistilBERT model. When it also failed, we proved that the problem was not the models, but the **low signal-to-noise**

ratio in the ticket description text. The language used across `Billing`, `General`, and `Technical` tickets was too generic and similar for any model to learn from.
- **The Successful Pivot:** Our solution was to engineer a better input feature (combining the `Ticket Subject` and description) and adopt a **hierarchical classification strategy**:
    1. **Gatekeeper Model:** A binary classifier that perfectly isolates "Account Access" tickets.
    2. **Specialist Model Analysis:** A second model that confirmed the remaining three categories could not be reliably separated.

This journey led directly to our recommendation of a hybrid AI + Human Triage workflow.

B. Semantic Understanding: Duplicate Detection & Search

For these tasks, we compared a traditional keyword-based method (TF-IDF) against a modern semantic method (Transformer embeddings).

- **TF-IDF Limitations:** The TF-IDF system was easily confused by generic boilerplate text and returned many irrelevant results.
- **Transformer Superiority:** The fine-tuned DistilBERT model was able to understand the *context* and *meaning* of the tickets. For duplicate detection, it identified semantically identical tickets with >99% confidence. For semantic search, it correctly identified thematically related past issues, even when keywords didn't match.

3. Final Recommendations & Strategic Roadmap

HelpNow has a clear, data-driven path to leveraging AI in its support operations. We recommend a phased approach:

Phase 1: Immediate Deployment (Short-Term)

1. **Deploy the "Gatekeeper" Model:** Immediately implement the `best-gatekeeper-model` to automate the routing of all "Account Access" tickets.
2. **Establish a Human Triage Queue:** Route all other tickets to a dedicated queue where agents perform the final categorization step. This balances automation with accuracy.
3. **Deploy the Duplicate Detection Function:** Provide agents with a tool based on our `find_embedding_duplicates` function to help them identify and merge redundant tickets.

Phase 2: Strategic Investment (Mid-Term)

1. **Invest in Transformer Infrastructure:** While our models can run on CPUs, performance will be significantly enhanced by investing in GPU resources for both training and real-time inference. This is a necessary step to scale the semantic search and other future AI features.
2. **Deploy the Semantic Search Engine:** Roll out the semantic search tool to the entire support team to improve resolution times and consistency.
3. **Improve Data Quality:** The single biggest lever for improving the AI is improving the data. Coach agents to write more descriptive and specific `Ticket Subjects`. Over time, this will generate better data that could eventually make the "Specialist" model viable, reducing the need for human triage.

Phase 3: Future R&D (Long-Term)

1. **Explore Multimodal Encodings:** The project plan's mention of hierarchical networks is forward-looking. As HelpNow grows, it may need to handle support requests that include screenshots or audio clips. Future work can focus on building models that can understand text, images, and audio simultaneously.
2. **Chatbot Integration:** The semantic search engine is the foundational technology for an internal or external chatbot. The model's ability to find relevant resolutions can be used to power automated answers to common questions.

4. Technical Appendix: Models, Metrics, and Artefacts

This section provides a detailed summary of the technical implementation, model performance, and key data artefacts from the project.

A. Data Preprocessing & Feature Engineering

- **Source Data:** 8,469 tickets with 17 initial columns.
- **Target Variable (`Category`):** Engineered from `Ticket Type` and `Ticket Subject` using the following mapping logic:
  - **'Account Access'**: Where `Ticket Subject` == 'Account access'.
  - **'Billing'**: Where `Ticket Type` was 'Billing inquiry', 'Refund request', or 'Cancellation request'.
  - **'Technical Issue'**: Where `Ticket Type` == 'Technical issue'.
  - **'General Query'**: Where `Ticket Type` == 'Product inquiry'.
- **Primary Text Feature:** A `combined_text` column was created by concatenating the raw `Ticket Subject` with a cleaned version of the `Ticket Description` (`Ticket Subject + " | " + Cleaned_Description`).
- **Text Cleaning Pipeline (`Cleaned_Description`):**
  1. Convert to lowercase.
  2. Remove URLs, numbers, and punctuation.
  3. Tokenize text.
  4. Remove standard English stopwords.
  5. Apply WordNet Lemmatization.

B. Task 1: Ticket Categorization - Performance Metrics

B.1. Baseline Model Performance (4-Class Problem)

A summary of the initial models trained on the `Cleaned_Description` alone:

Model	Text Representation	Weighted F1-Score	Key Observation
Logistic Regression	Bag-of-Words	**0.43**	Biased towards 'Billing', but attempted all classes.
Naive Bayes	TF-IDF	0.41	Collapsed and only predicted 'Billing'.
LSTM w/ GloVe	Word Embeddings	~0.06	Collapsed and failed to converge.

B.2. Advanced Model Performance (4-Class Problem)

The DistilBERT model, trained on `combined_text`, also struggled with the 4-class problem due to data limitations.

- **Accuracy:** 62.46%
- **Weighted F1-Score:** 0.48
- **Classification Report:**

Category	Precision	Recall	F1-Score
Account Access	**1.00**	**1.00**	**1.00**
Billing	0.60	1.00	0.75
General Query	0.00	0.00	0.00
Technical Issue	0.00	0.00	0.00

- **Key Insight:** This result proved a hierarchical approach was necessary, as the model perfectly solved for "Account Access" but defaulted to "Billing" for all other categories.

B.3. Final Hierarchical Model Performance

This is our recommended production system.

1. Gatekeeper Model (`best-gatekeeper-model`)

- **Task:** Binary classification ('Account Access' vs. 'Other').
- **Accuracy: 1.0000**
- **Weighted F1-Score: 1.00**
- **Confusion Matrix:**

	Predicted: Other	Predicted: Account Access
Actual: Other	**796**	0
Actual: Account Access	0	**51**

- **Conclusion:** The Gatekeeper is a near-perfect classifier, validating the first stage of our pipeline.

2. Specialist Model (`best-specialist-model`)

- **Task:** 3-Class classification on remaining tickets.
- **Result:** The model collapsed and predicted 'Billing' for all inputs.
- **Conclusion:** This proved that the remaining categories are not textually separable with the current data, validating our recommendation for a human Triage Queue.

C. Task 2: Duplicate Ticket Detection - Performance

- **Baseline (TF-IDF):** Produced low-confidence scores (~0.5) and returned irrelevant results based on shared boilerplate text.
- **Advanced (Transformer Embeddings):**
  - **Method:** Cosine similarity on embeddings generated from the fine-tuned `best-gatekeeper-model`.
  - **Performance:** Identified semantically identical tickets with extremely high confidence scores (**>0.99**). Successfully grouped tickets by underlying meaning, ignoring keyword differences. This is the recommended approach.

D. Task 3: Semantic Search - Performance

- **Method:** Generated an index of embeddings for all historical `Resolution` texts. Cosine similarity was used to match a new ticket's embedding to the resolution index.
- **Performance:** The system successfully identified thematically relevant past solutions. For a query about "Product compatibility," the top 5 results were resolutions from past tickets about `Installation support`, `Data loss`, `Network problem`, `Product setup`, and `Product compatibility`.
- **Key Insight:** The system works semantically, clustering tickets by topic. Its effectiveness in a production environment is directly tied to the quality of the written resolution text.

E. Final Deployed Artefacts

- **`best-gatekeeper-model`:** A saved Hugging Face model directory. This is the core of the classification system.
- **`predict_ticket_category()` function:** A Python function that encapsulates the full hierarchical logic (including text cleaning) and returns a final prediction of either "Account Access" or "Triage Required (Billing/General/Technical)". This is the primary function to be integrated into HelpNow's systems.
- **`find_embedding_duplicates()` function:** A Python function that takes a ticket ID and returns a list of potential duplicates.
- **`search_knowledge_base()` function:** A Python function that takes a raw ticket text and returns a list of the most relevant historical resolutions.