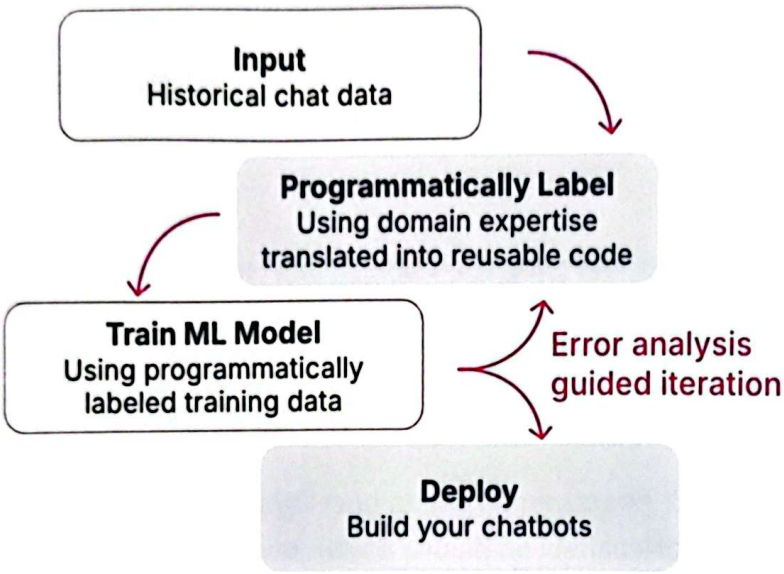
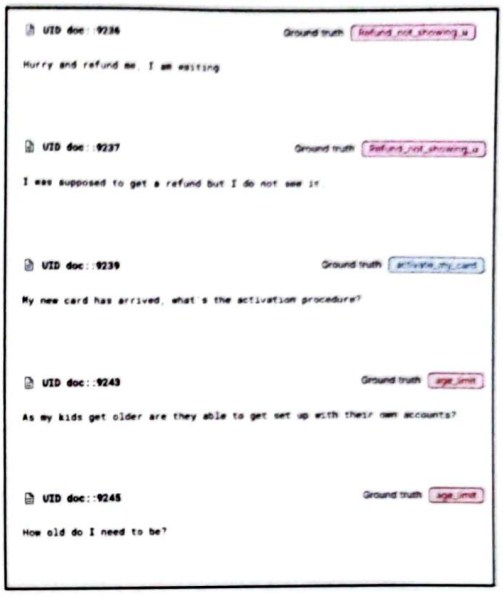
Faster chatbot intent classification with Snorkel





In this Snorkel Flow workshop, we'll classify user intent from customer conversations. The dataset contains 13,000 unlabeled data points and 300 labeled examples across 77 distinct intent classes. This workshop is based on a real-world customer use case involving 500 intent classes, where the objective is to customize LLMs and ultimately deploy a distilled, more efficient model. Although finance chatbots might appear to be a standard application, we'll soon explore how LLMs alone often fall short in capturing the unique organizational knowledge required for the systems and categories specific to banks.

Snorkel Flow

FM-First Chatbot Intent Classification Workshop

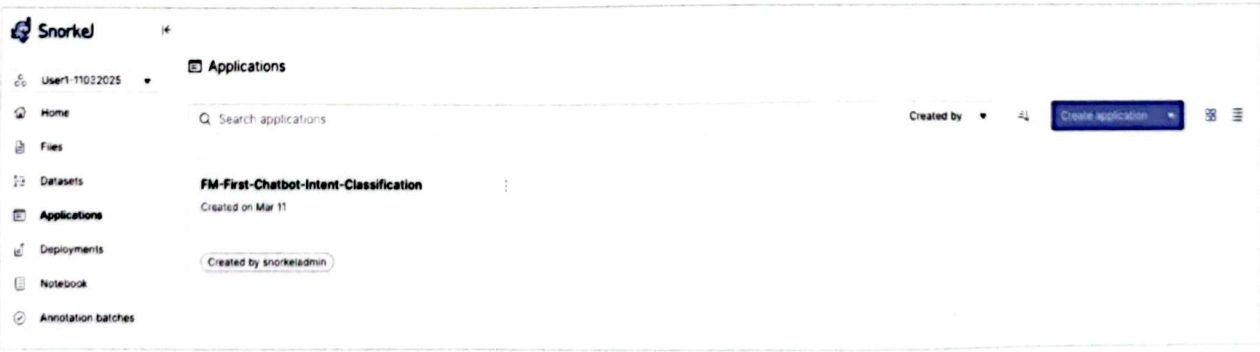
Instructions

1. Login

Access the Snorkel Flow platform using the credentials provided.

2. Verify Workspace

After logging in, you will land on the home screen. Confirm that the top-left corner shows your designated workspace, which should be identical to your login username. If it displays "default", use the drop-down menu to select the workspace associated with your credentials.

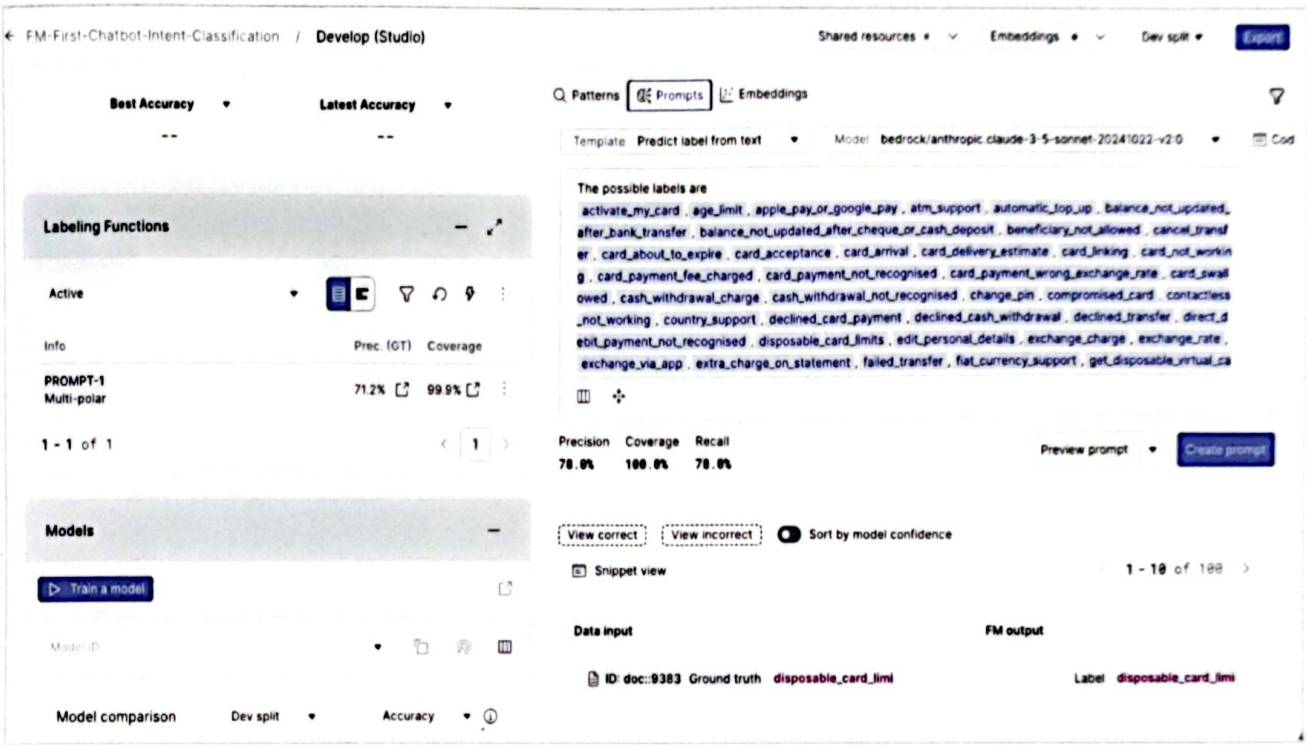


3. Navigate to the Application

On the left hand side, press **Applications**, and select **FM-First-Chatbot-Intent-Classification**.

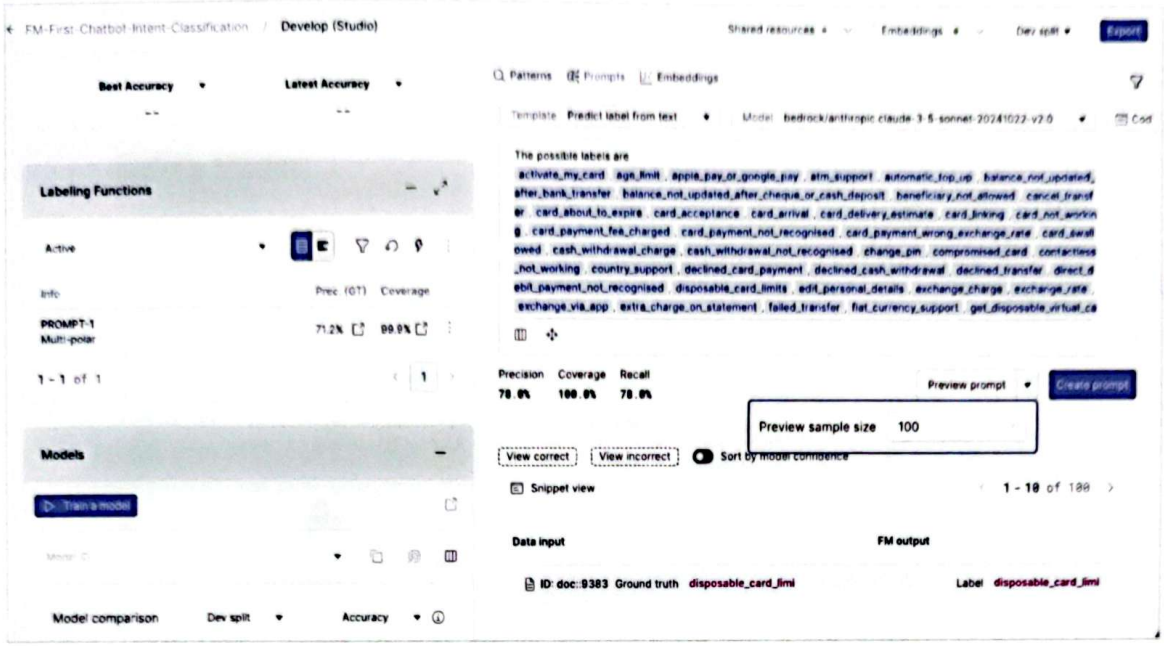
4. Begin with Foundation Model (FM)-based Intent Classification

Since there are 77 classes, let's start our classification efforts with a foundation model. On the Studio page, select **Prompts.**



5. Previewing FM-based Prompt Outputs

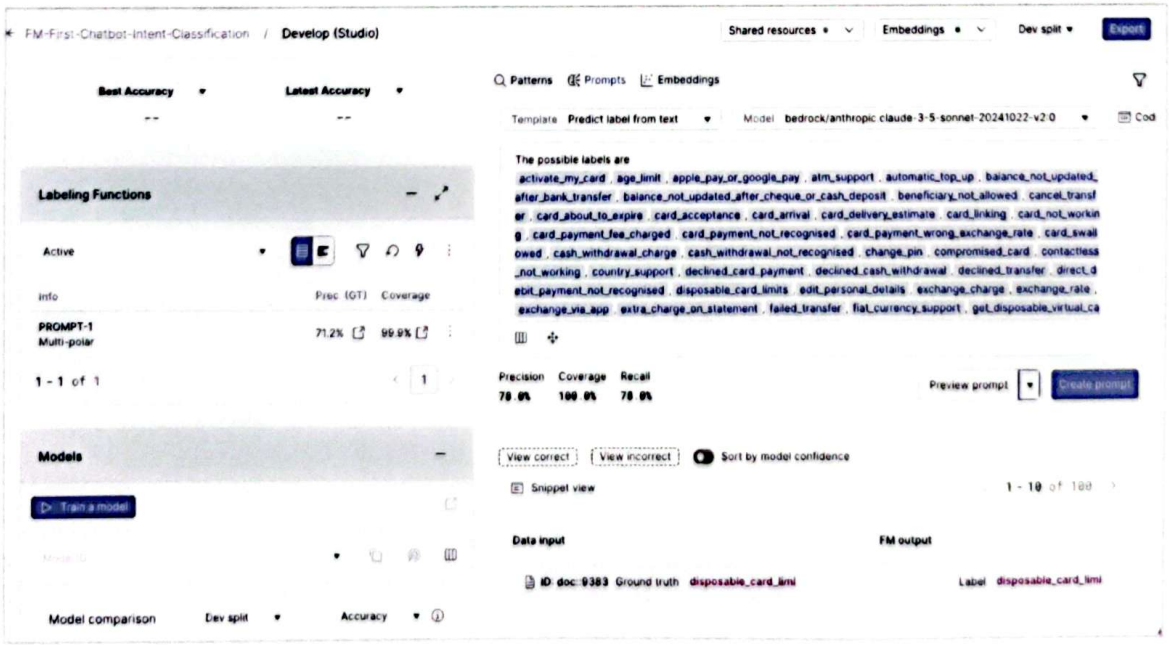
We will utilize the default prompt within the prompt field, applying the **bedrock/anthropic.claude sonnet 3.5** model for our language model predictions. Since running our entire dataset over an LLM endpoint requires a significant number of API calls, let's preview our prompt on just 100 predictions.



Do not press Create Prompt. The model has already been prerun.

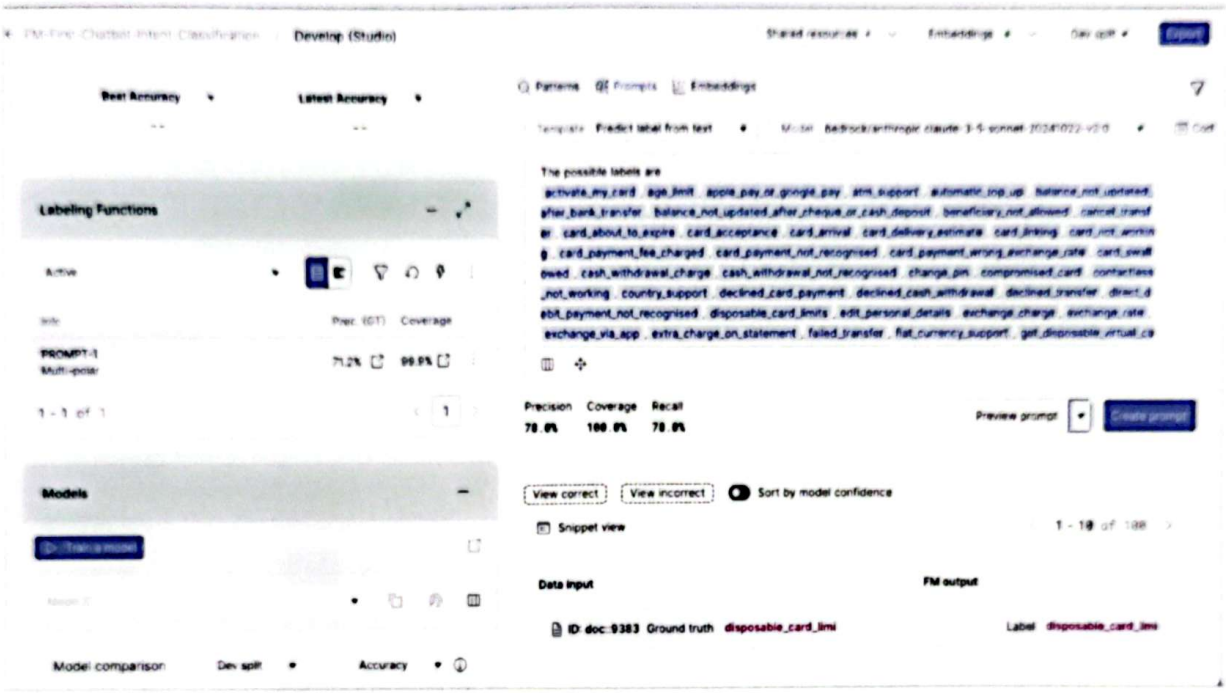
6. Reviewing Preliminary Metrics

The FM prediction metrics for this set of 100 data points yields a 78% precision and 78% recall. The next step here would be to press "Create Prompt" to run the FM on the dev and training splits. For efficiency, we have pre-populated the Claude Sonnet 3.5 model as a Labeling Function. We see that the precision on the dev split decreases to 71.2%.



7. Train our First Model

Let's train our first model, which distills the LLM into one small model. In the **Models** section, click on **Train a Model**

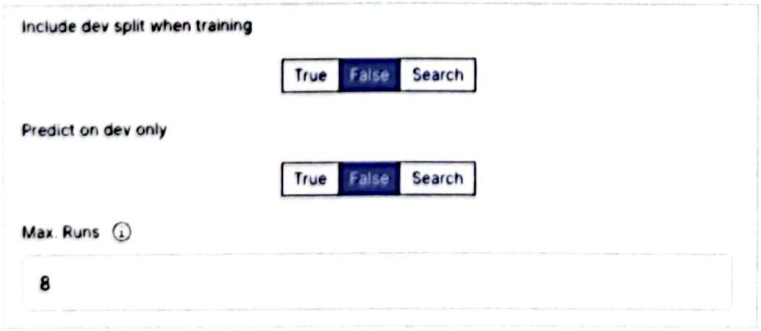


Within the training model pop-up, keep the model name **"model-1"**. In **Input Fields** uncheck "index" so that **"text"** is the only column used for model input:

Feel free to explore how the Feature Libraries, Train Options, and Model Options can be customized for model training. However, please ensure that your final configuration settings align with the ones shown below to maintain consistency throughout the rest of the demo:

Feature Libraries

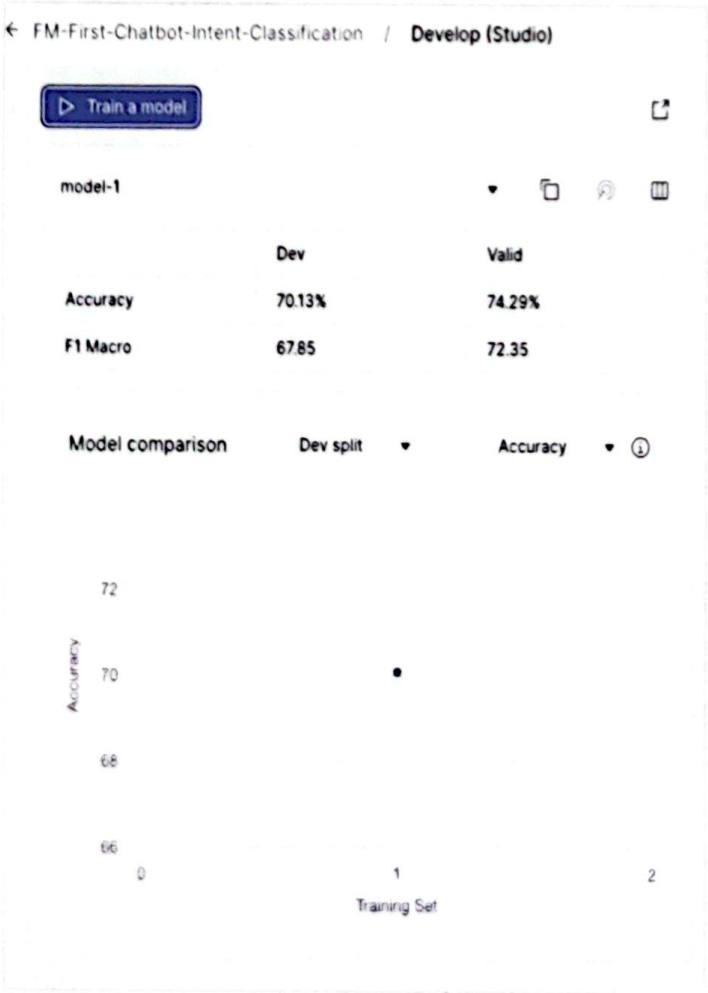
Train Options



Model Options

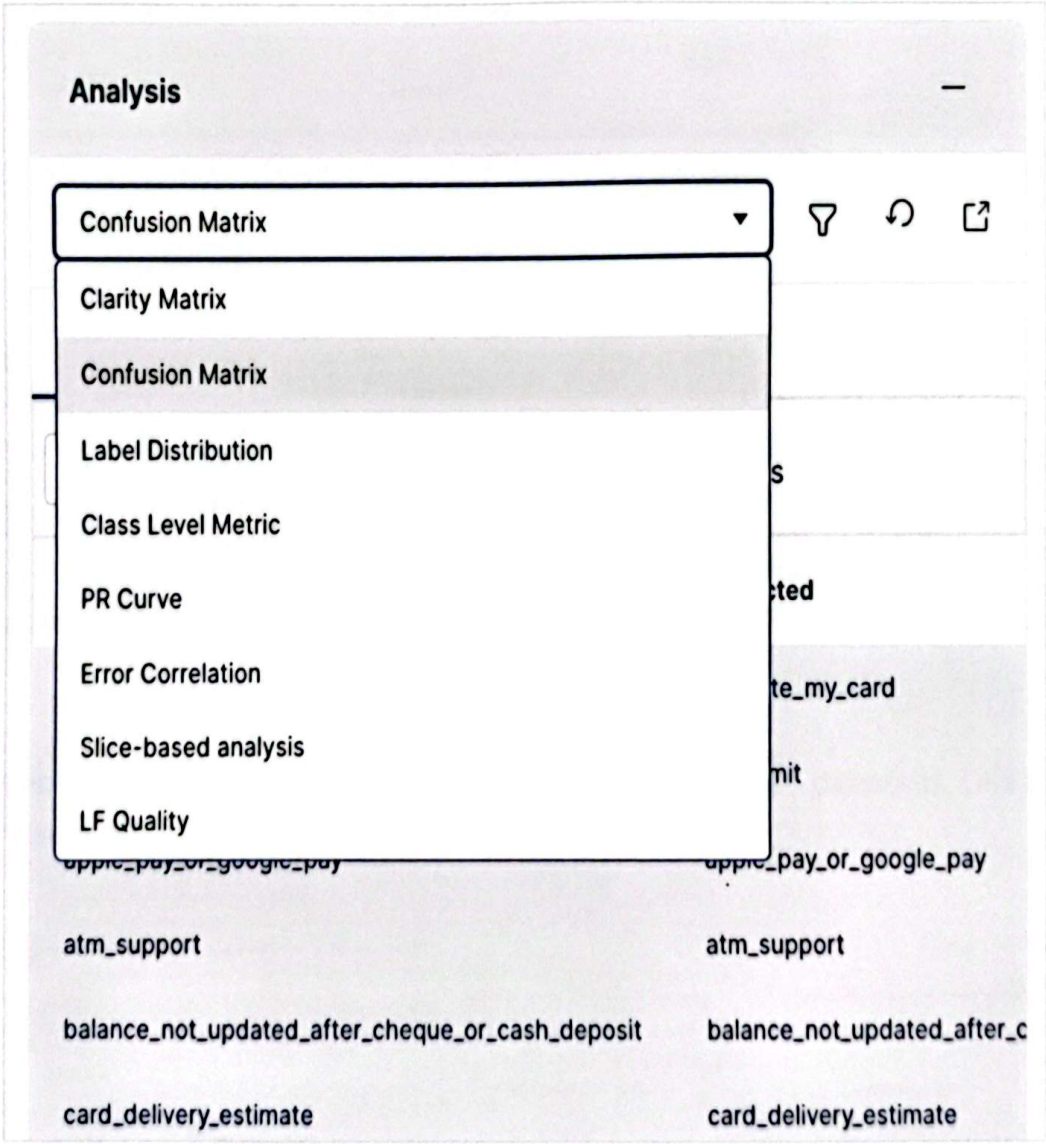
```
{
  "classifier": {
    "classifier_cls": "LogisticRegression",
    "classifier_kwargs": {
      "C": 10,
      "penalty": "l2",
      "solver": "saga",
      "random_state": 123
    }
  },
  "text_vectorizer": {
    "vectorizer_cls": "TfidfVectorizer",
    "vectorizer_kwargs": {
      "ngram_range": [
        1,
        2
      ],
      "max_features": 250000,
      "lowercase": false
    }
  }
}
```

Once complete, press **Train custom model.** The training process may take a few minutes, after which the model's metrics for all splits will be available.

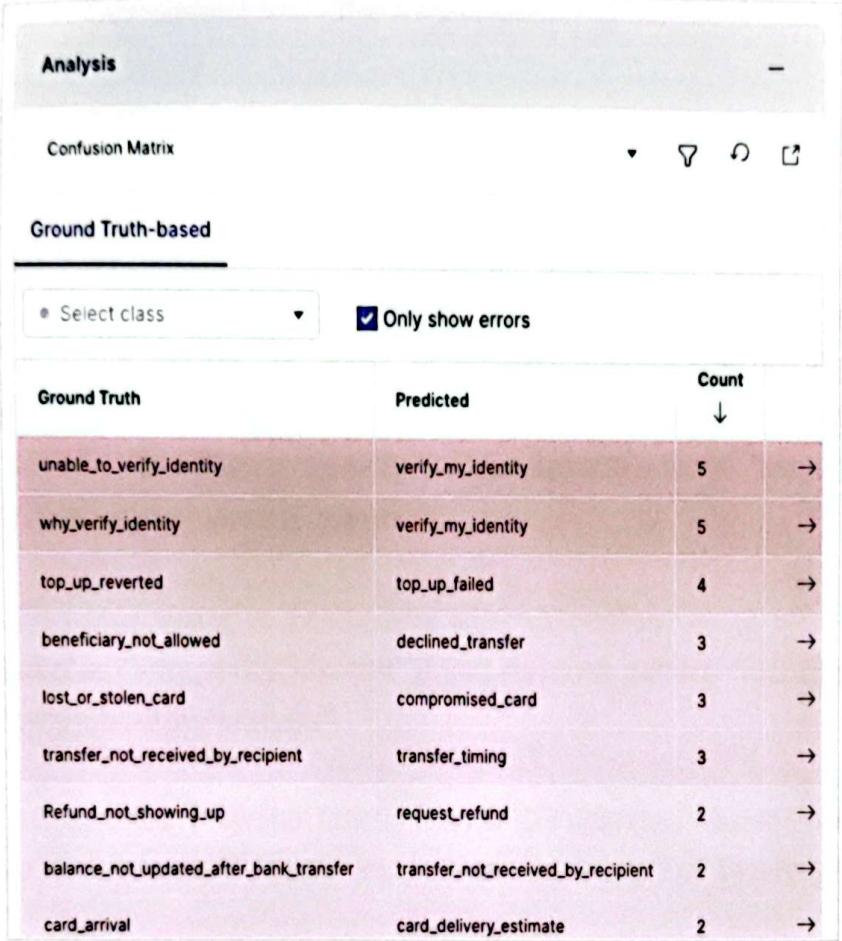


Error Guided Analysis with Confusion Matrix

Go to the **Analysis** panel and navigate to the **Confusion Matrix**, where we will begin to iterate on model-1 using Snorkel Flow's error-guided analysis.



Let's filter for only instances where there were errors. Apply the **"Only show errors"** filter to focus on instances where errors occurred. Here, Snorkel Flow automatically sorts based on which classes have the highest error count. By clicking on the right arrow in the rightmost column, the data viewer will filter for specific class confusion pairs. **Let's click on the second row "why_verify_identity / verify_my_identity":**



Analysis

Confusion Matrix

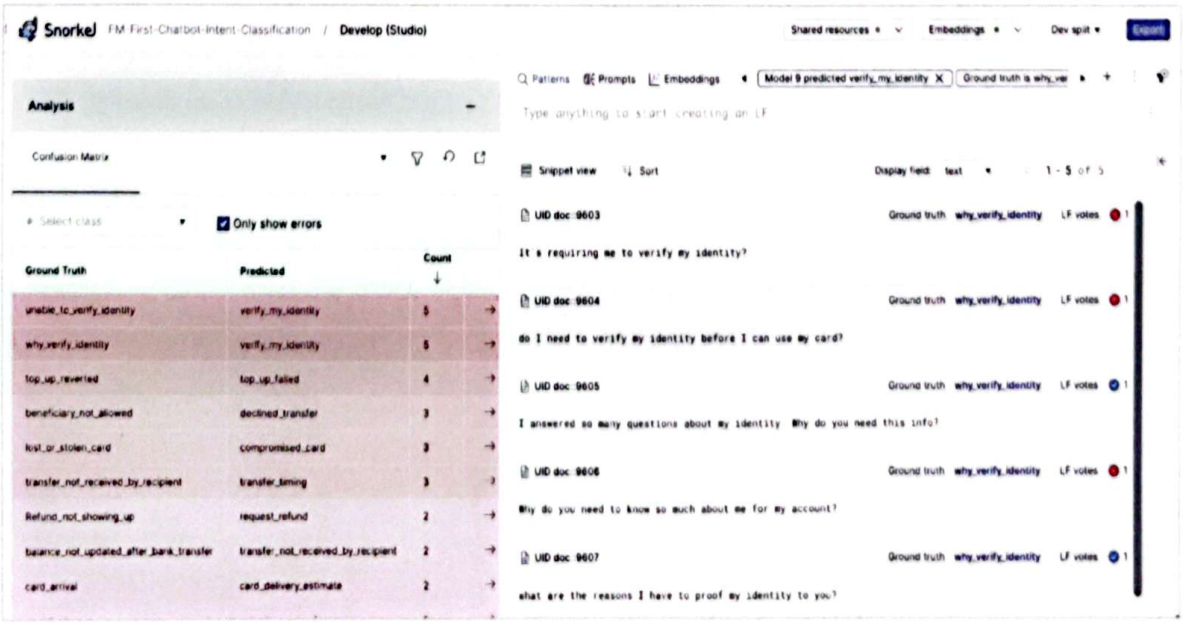
Ground Truth-based

Select class

Only show errors

Ground Truth	Predicted	Count ↓	
unable_to_verify_identity	verify_my_identity	5	→
why_verify_identity	verify_my_identity	5	→
top_up_reverted	top_up_failed	4	→
beneficiary_not_allowed	declined_transfer	3	→
lost_or_stolen_card	compromised_card	3	→
transfer_not_received_by_recipient	transfer_timing	3	→
Refund_not_showing_up	request_refund	2	→
balance_not_updated_after_bank_transfer	transfer_not_received_by_recipient	2	→
card_arrival	card_delivery_estimate	2	→

Once clicked, the data viewer will refresh and filter for just the 5 datapoints. Let's click on the **"Patterns"** section and select **"Snippet View"**:



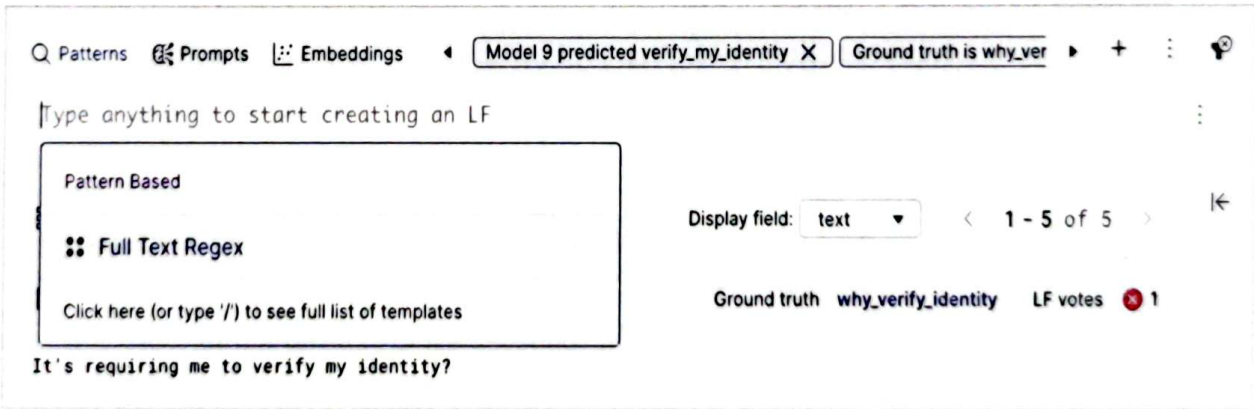
8. Developing our First Regex Labeling Function (LF)

Scroll through the four data points and brainstorm how we could create a labeling function (LF) that accurately labels only these specific examples.

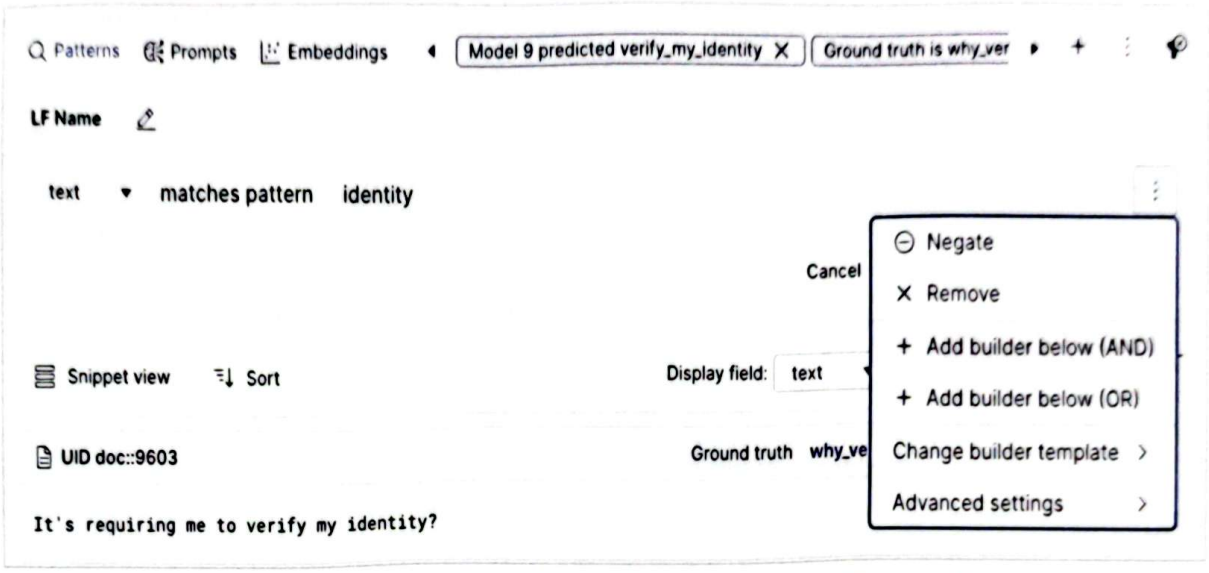
As we review the data, we notice that the word "identity" appears in almost all data points. We could create a labeling function that looks for the word "identity" and labels these instances as "why_verify_identity." However, it's likely that "identity" also appears in other contexts and labels, so we can refine the LF by adding additional criteria to make the function more nuanced. For instance, we could include terms like "why", "about", "before", "you", which we also observe in these examples.

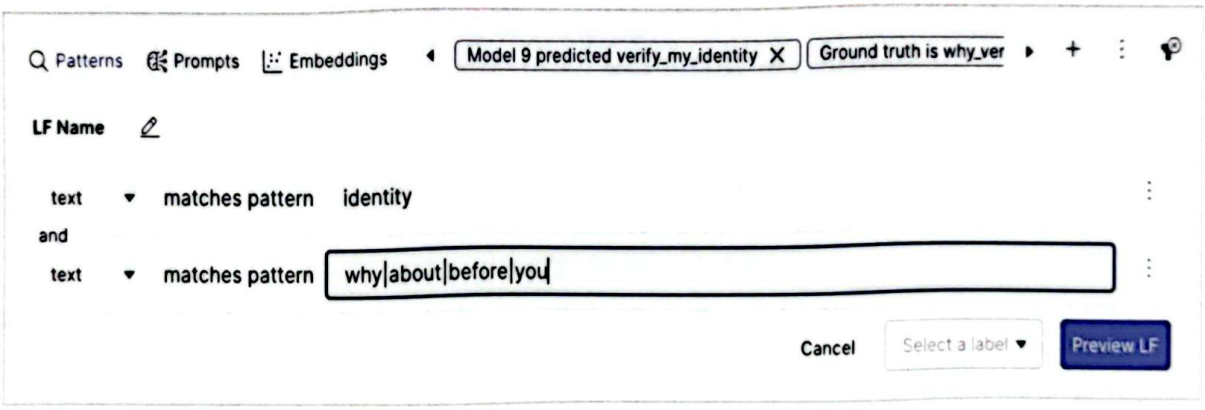
This would lead us to a compound labeling function:
Label as "why_verify_identity" if text matches pattern "identity" AND text matches pattern "why|about|before|you".

In Regex, the "|" symbol functions as an OR operator. We can implement this LF by entering the patterns into the "Patterns" text field and selecting "Full Text Regex.":

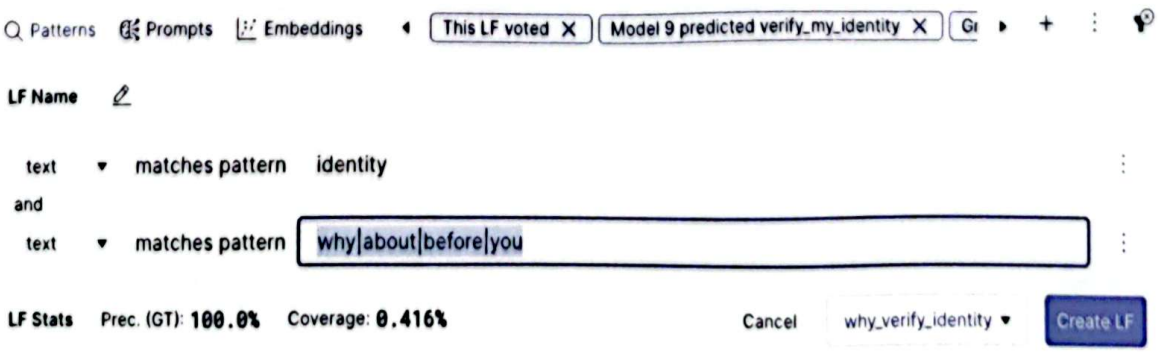


Let's populate the first field. Type **"identity"** in the first line. To add the second clause of our labeling function, press the three dots on the right side, and press "Add another builder (AND)". Add another Full Text Regex with **"why|about|before|you"**.

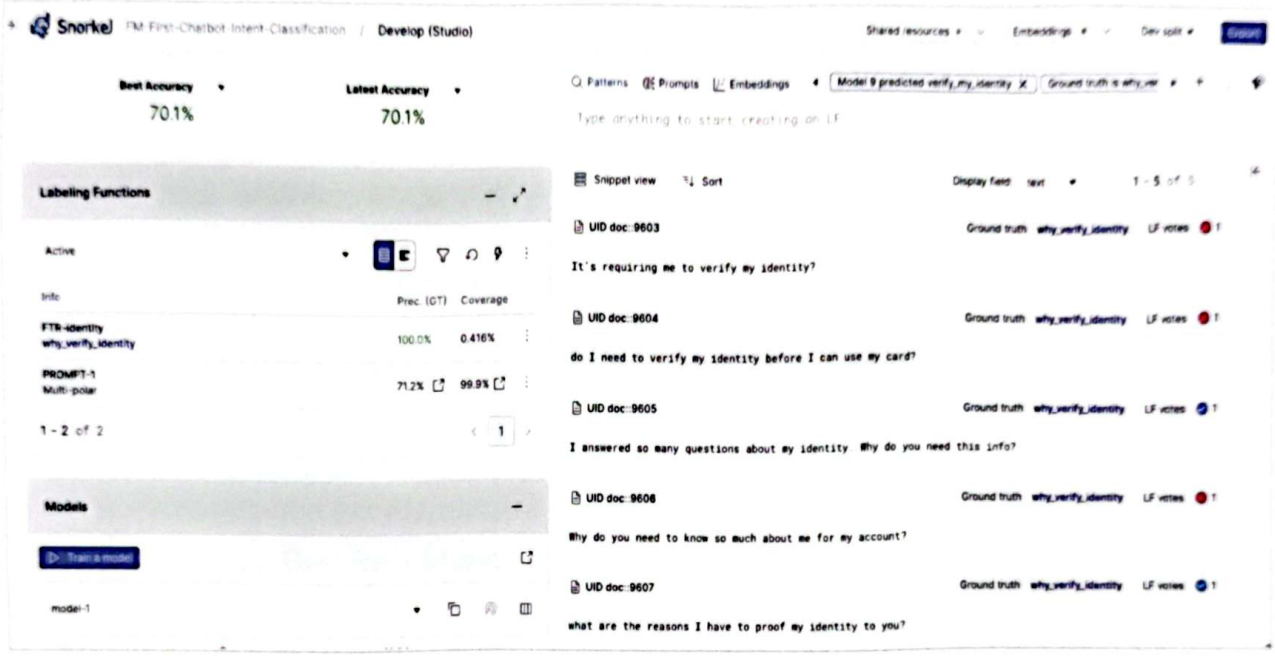




Click on "Preview LF". There is no need to manually select a label—Snorkel will automatically assign the most optimal label. In this case, Snorkel assigns the label "why_verify_identity," which aligns with our desired class. We can see that this LF achieves 100% precision and covers 0.416% of our dataset. Once satisfied, click "Create LF" to finalize the LF:

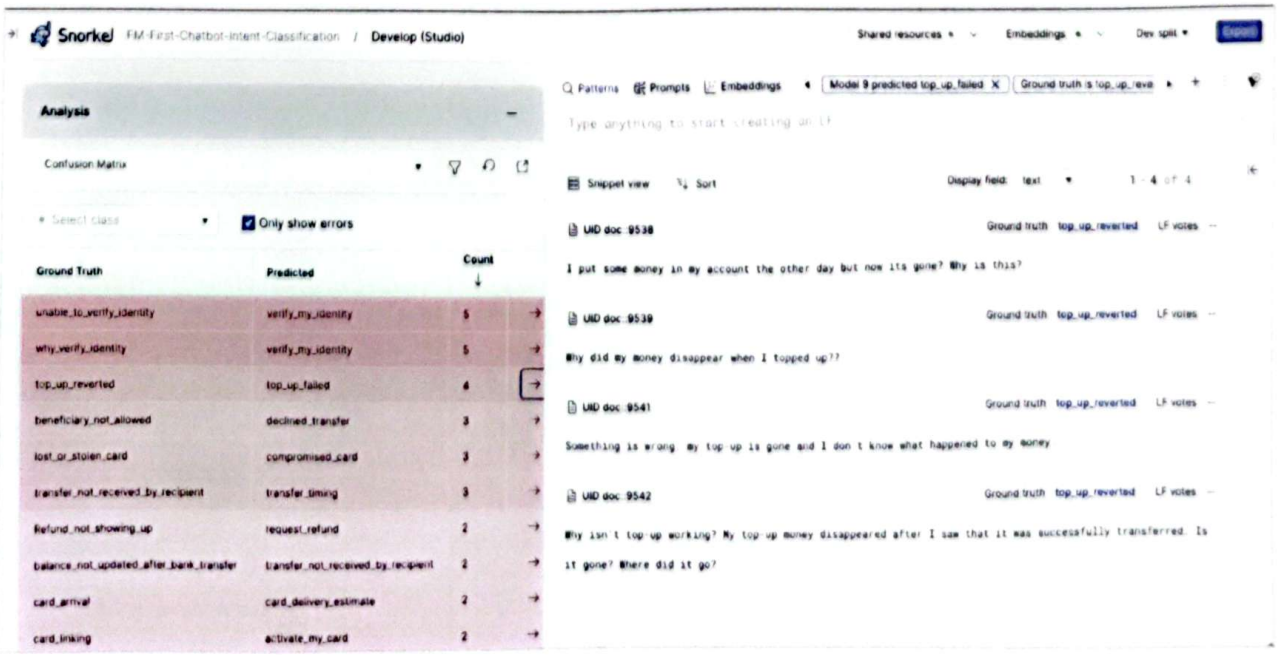


We have just finished creating our first LF! In now appears in the list of labeling functions on the left hand side:

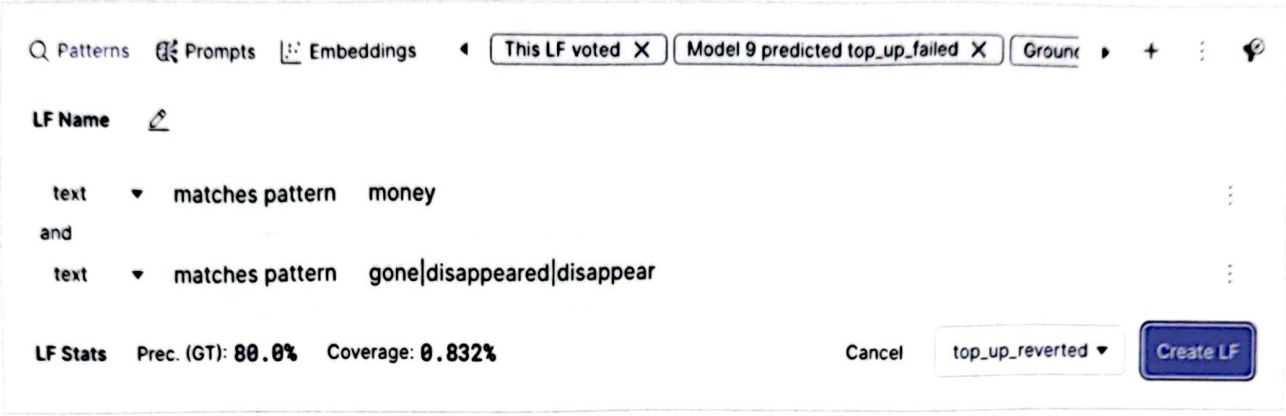


9. Developing our Second Regex Labeling Function

Now, let's create another LF using the Confusion Matrix. This time, **let's click on "top_up_reverted" / "top_up_failed":**

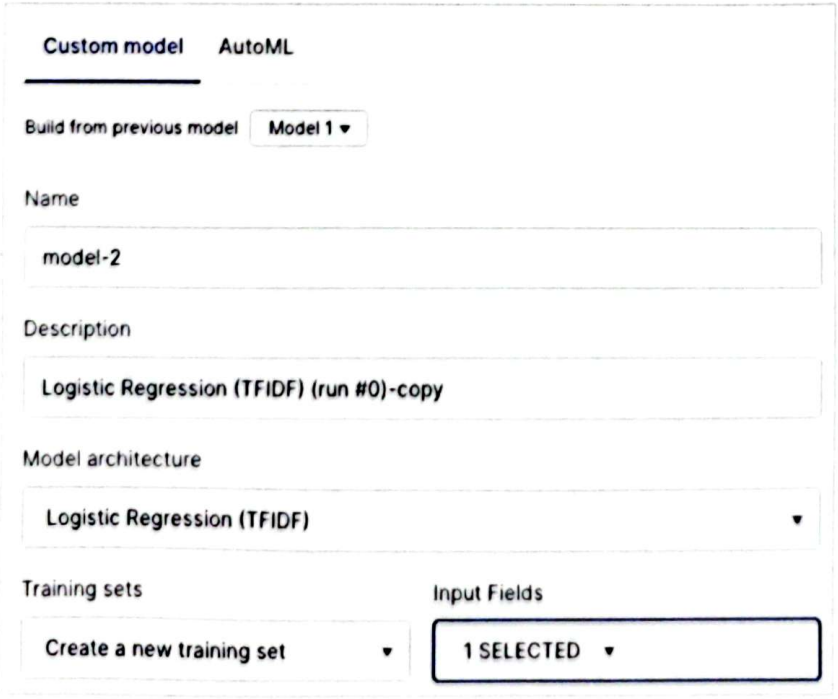


Using the same steps described above in **Step 9**, we can create the following compound LF that looks for the word **"money"** AND **"gone|disappeared|disappear"**:

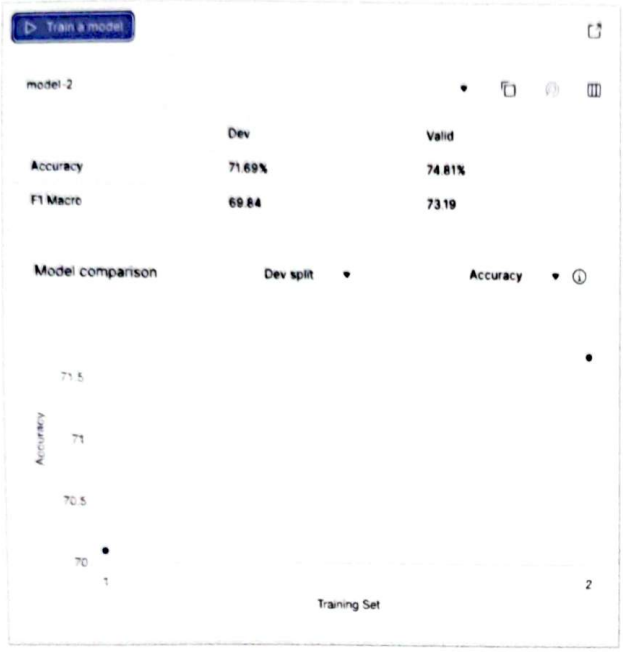


10. Iterating on Our First Model

Let's train an updated model. Return to the **Models** panel and click on **Train a model.** Select **Build from previous model** and choose **"model-1."** Name this new model **"model-2."** Under "Training Sets," select only **Create a new training set** as we've added additional LFs, which will prompt the label model to generate an updated set of labels for our unlabeled data. Leave all other parameters unchanged.

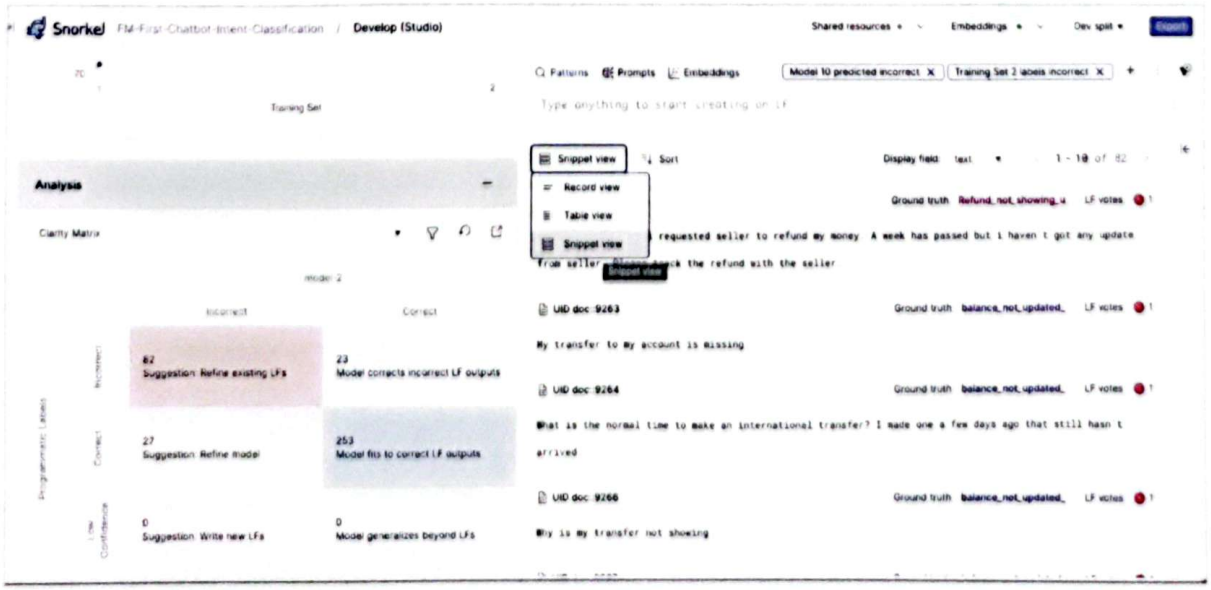


Press **Train custom Model.** After training, we expect the following improved metrics:



11. Error Guided Analysis with Clarity Matrix

Let's proceed with another iteration, this time exploring the Clarity Matrix and Snorkel's embedding-based LFs. Navigate to the **Clarity Matrix** under the Analysis panel. This matrix displays the number of data points where there is agreement or disagreement between the Label Model and the trained distilled model. Click on the **Incorrect / Incorrect** cell to filter the Data Viewer, and then switch to **Snippet View** so we can view multiple data points simultaneously:



Patterns Prompts Embeddings Model 10 predicted incorrect X Training Set 2 labels incorrect X

Type anything to start creating an LF

Snippet view Sort Display field: text 1 - 10 of 82

UID doc::9263 Ground truth balance_not_updated_ LF votes 1

My transfer to my account is missing.

UID doc::9264 Ground truth balance_not_updated_ LF votes 1

What is the normal time to make an international transfer? I made one a few days ago that still hasn't arrived.

UID doc::9266 Ground truth balance_not_updated_ LF votes 1

Why is my transfer not showing

UID doc::9267 Ground truth balance_not_updated_ LF votes 1

Hello I have a UK account. Can you please check if everything is alright with it? I made a transfer a couple hours ago but it doesn't show up yet.

In records 2-5, we observe that **"balance_not_updated_after_bank_transfer"** is mispredicted by both the label model and the end model. Let's click into these data points and brainstorm how to create an LF that can correct these misclassifications, using the same methodology outlined in Step 9. We can design the following LF that looks for the word **"transfer"** AND **"missing|show|arrived"**:

Patterns Prompts Embeddings This LF voted X Model 10 predicted incorrect X Training S

LF Name

text matches pattern transfer

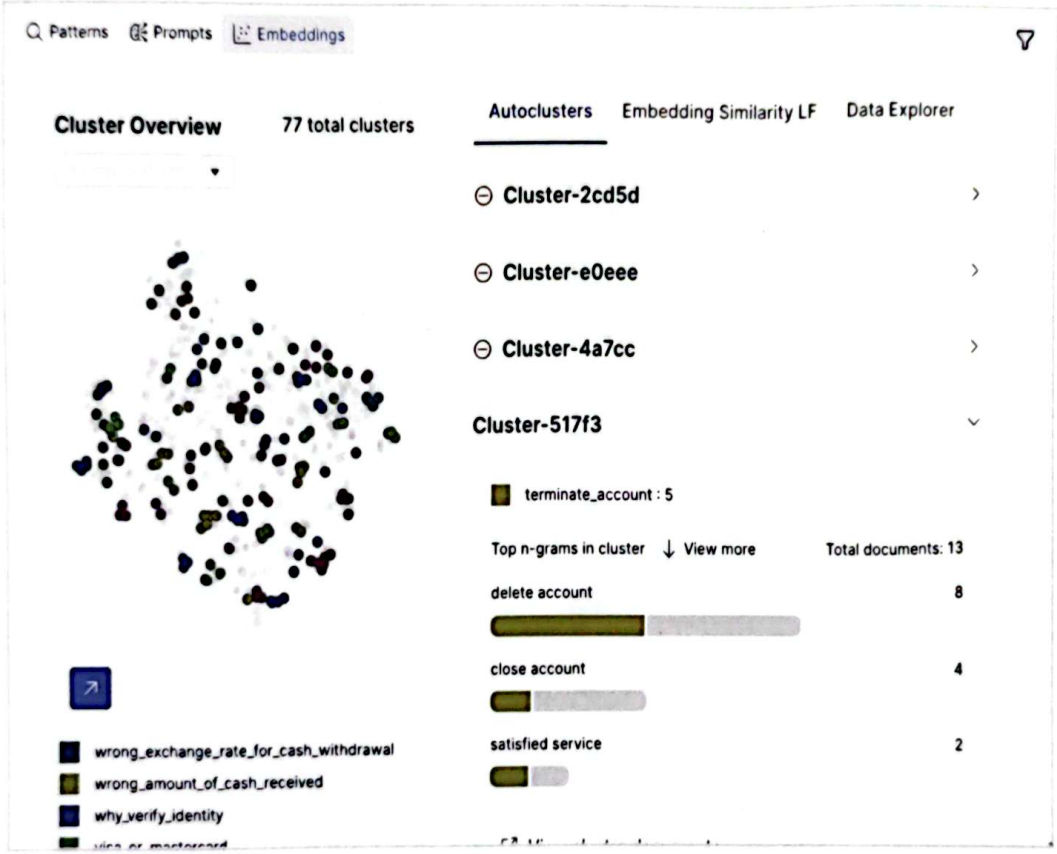
and

text matches pattern missing|show|arrived

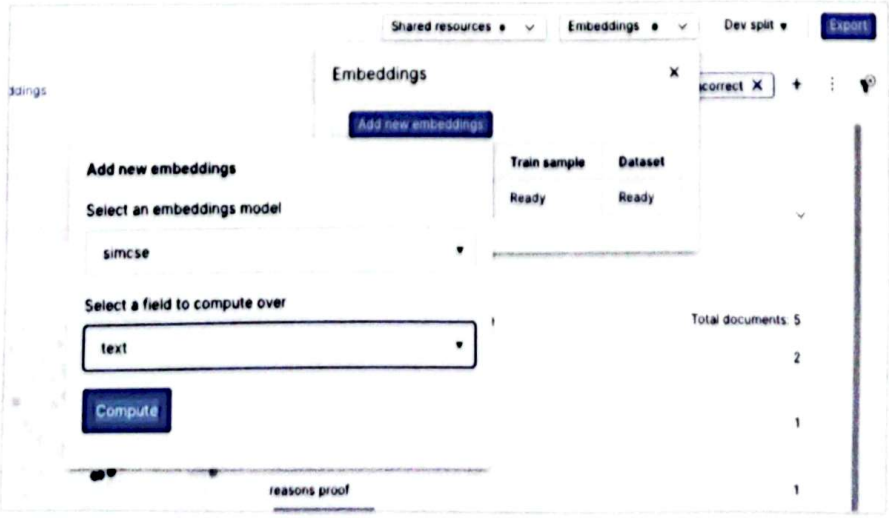
LF Stats Prec. (GT): 100.0% Coverage: 0.624% Cancel balance_not_updated_after_bank_transfer Create LF

12. Creating an Embedding-based LF

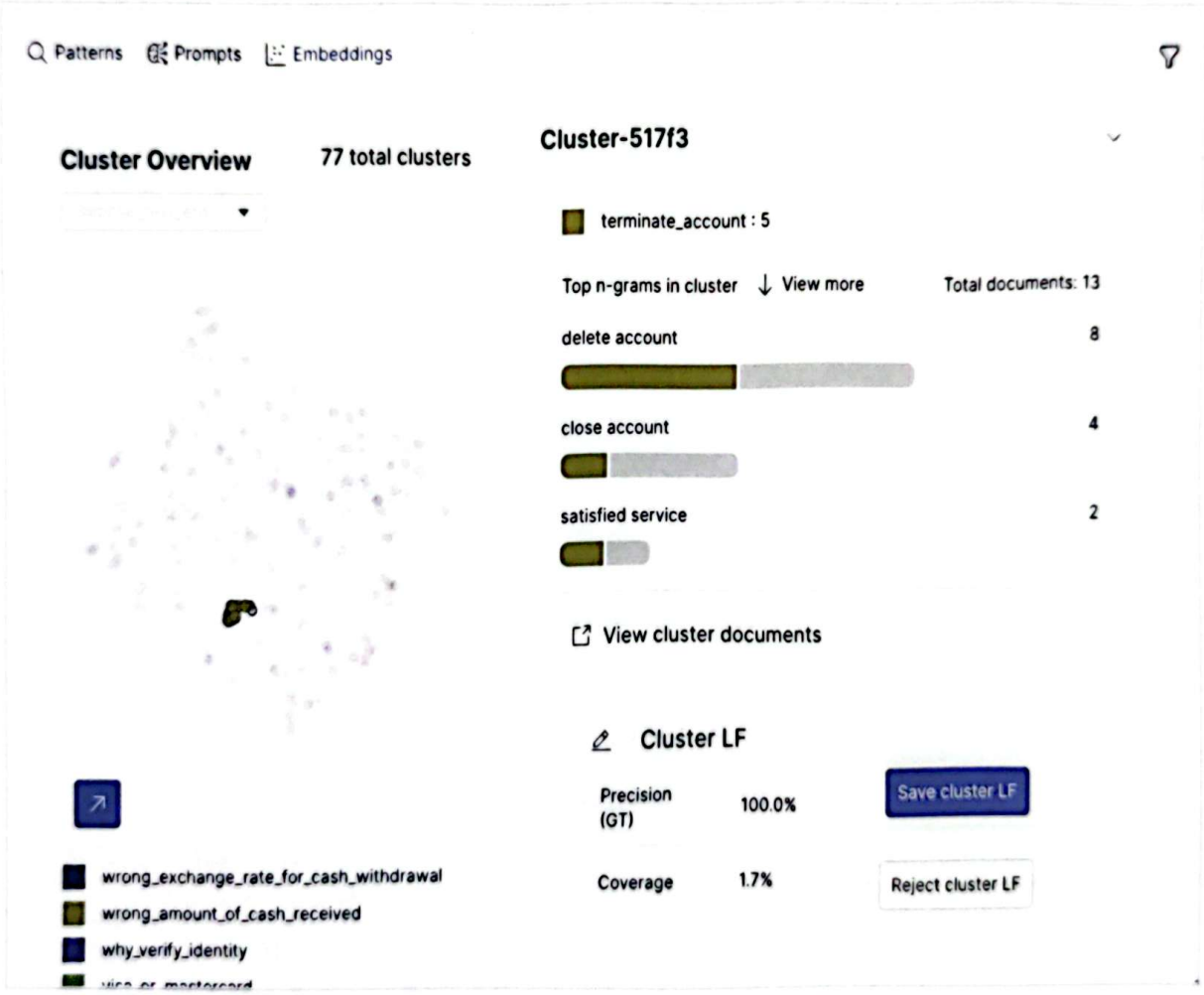
Now, let's create an Embeddings based LF. Clear the filters in the right hand corner and press the Embeddings button. This may take a second to load.



If there are no embeddings, press "Embeddings" on the top right of the screen, and press "Add New Embeddings". Select "simcse" embeddings model computed over the "text" field:



The Embeddings page uses semantic embeddings to identify clusters of semantically similar text that can be leveraged as LFs. Here, we see an auto-suggested cluster for "terminate_account". Note that the clusters shown in the visualization may not be close together because it is a projection of high-dimensional space onto 2-dimensions.



Press **"Save cluster LF"** to add this cluster as a labelling function.

Note: You can explore the cluster and the documents in it by clicking "View Cluster Documents". You can also create your own cluster in the "Data Explorer" page by using the "Data Drag Explorer" tool.

Cluster Data Viewer: Cluster-517f3

Snippet view

1 - 10 of 13

UID doc::5676 — Ground truth UNKNOWN — LF votes 2

I need to delete my account, please.

UID doc::5706 — Ground truth UNKNOWN — LF votes 2

your company is awful, i would like to delete my account please

UID doc::5717 — Ground truth UNKNOWN — LF votes 2

How to I close my account?

UID doc::5726 — Ground truth UNKNOWN — LF votes 2

How can I close my account?

UID doc::5730 — Ground truth UNKNOWN — LF votes 2

I am trying to delete my account.

Patterns | Prompts | Embeddings

Cluster Overview

- wrong_exchange_rate_for_cash_withdrawal
- wrong_amount_of_cash_received
- why_verify_identity
- visa_or_mastercard
- virtual_card_not_working

Autoclusters | Embedding Similarity LF | Data Explorer

Data Drag Explorer | Hierarchical Cluster Explorer

Selection with 68 points

- beneficiary_not_allowed : 4
- balance_not_updated_after_bank_transf : 3

Top n-grams in cluster ↓ View more — Total documents: 68

long does	6
transfer declined	6
does transfer	5

View cluster documents

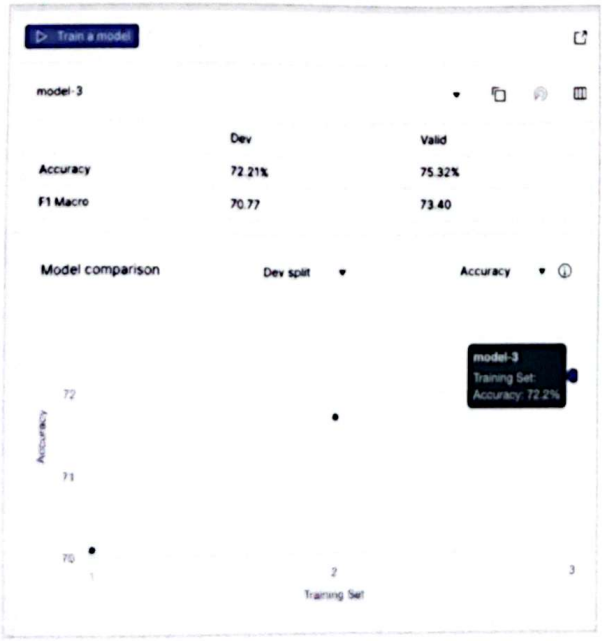
Create a cluster LF — Select a label — Preview LF

Snorkel

13. Iterating on Our Second Model

Now that we have two additional LFs, let's train an updated model. Return to the **Models** panel and click on **Train a model.** Select **Build from previous model** and choose **"model-2."** Name this new model **"model-3."** Under "Training Sets," select only **Create a new training set** as we've added additional LFs, which will prompt the label model to generate an updated set of labels for our unlabeled data. Leave all other parameters unchanged.

Upon completion of training, we see improved metrics across our Dev and Valid splits for both accuracy and F1:



Feel free to continue to iterate and create more labeling functions. Note that this process yields a model that is an MLflow object that is ready to be deployed:

Models

> Advanced options

ID	Model Name	Runtime	Description	Timestamp (UTC)	Training Set	
11	model-3	3 minutes ago	Logistic Regression (TFIDF) (run #0)-copy (run #0)-copy (run #0)	2025-03-14 03:22:09	3	Commit

Show Test

Split	Accuracy	F1 Macro
dev	72.2%	70.8%
valid	75.3%	73.4%

10	model-2	18 minutes ago	Logistic Regression (TFIDF) (run #0)-copy (run #0)	2025-03-14 03:06:43	2	Commit
9	model-1	about 1 hour ago	Logistic Regression (TFIDF) (run #0)	2025-03-14 02:37:24	1	Commit