Recombo.AI

DS/ML Engineer Test

Type: “Job Title”, Updated at: October 2025

Overview

You will build and evaluate a small end-to-end workflow that involves synthetic data generation, model training, and simple analysis. Each step should naturally connect to the next one to form a coherent workflow.

Summary of tasks:

- **Task 1:** Generate a synthetic dataset with realistic job title pairs following at least five rules.
- **Task 2:** Train a binary classification model to predict whether a pair should be marked verified or discrepancy.
- **Task 3:** Analyze and interpret model errors, identifying potential patterns and proposing ideas for detecting issues automatically.

Guidelines:

- **Language:** Python
- **Tools:** Any framework or libraries (open-source or commercial)
- **Delivery:** Jupyter notebook and a dataset in CSV format
- **Deadline**: 1 week

Task 1: Generate a Synthetic Dataset

Create a dataset large enough for model training that contains job title pairs representing two sources that may or may not agree. Each row should include:

```
None
case_id, applicant_job_title, official_record_job_title, status, rule
C0001, senior software engineer, software engineer, discrepancy, Rule 4 (Hierarchy / seniority
difference)
C0002, receptionist at murphy oil, receptionist, verified, Rule 2 (Normalization)
C0003, n/a, plumber, verified, Rule 1 (Missing Title)
C0004, data scientist, associate, verified, Rule 5 (Departmental / generic vs specific)
C0005, software engineer, account executive, discrepancy, Rule 7 (Different job family)
C0006, full time, software engineer, verified, Rule 6 (Invalid title)
```

These examples demonstrate how each rule might appear in the dataset. See the rule list (next page) for reference. You should implement at least **five** of these rules (or more) to produce realistic examples.

Where `status` is **verified** or **discrepancy**.

Aim for a few thousand rows if possible, ensuring enough examples for training.

You may source or construct base titles from public data such as:

- **Kaggle Resume Dataset** – public dataset with job titles and categories: https://www.kaggle.com/datasets/gauravduttakiit/resume-dataset
- **O*NET Job Titles** – official U.S. Department of Labor occupation database: https://www.onetcenter.org/database.html
- **Indeed Job Titles Dataset (Kaggle)** – scraped job titles and descriptions: https://www.kaggle.com/datasets/promptcloud/indeed-job-posting-dataset/data

Use these sources or similar open datasets to seed your job title list before generating synthetic pairs using your rules. You can optionally use ChatGPT or similar AI tools to generate a list of N job titles for your dataset. Please document the source you used for dataset generation, whether it's an online dataset, API, or AI-generated list.

Deliverable: `dataset.csv` - this will serve as the training data for the next task.

Recombo.AI

Job Title Verification Rules

#	Rule	Description	Example Outcome
1	**Missing Title → verified**	If either title is null, “n/a”, or “missing”, mark as verified.	“N/A” vs “Plumber” → verified
2	**Normalization**	Remove company names, locations, and incomplete words before comparing.	“Receptionist at Murphy Oil” → “Receptionist”
3	**Exact / near match → verified**	If cleaned titles are identical or very close (e.g., ≥ 0.9 similarity), mark verified.	“Software Engineer” vs “Engineer, Software” → verified
4	**Hierarchy / seniority difference → discrepancy**	Same base role, but one includes seniority (Junior, Senior, Lead, I/II/III).	“Software Engineer” vs “Senior Software Engineer” → discrepancy
5	**Departmental / generic vs specific → verified**	One side is a department or generic term (Sales, Associate, Specialist).	“Sales” vs “Sales Manager” → verified
6	**Invalid title → verified**	Title is a company name or employment type (“Meta Inc.”, “Full Time”).	“Software Developer” vs “Full Time” → verified
7	**Different job family → discrepancy**	Clearly different job types (Tech vs HR).	“Software Engineer” vs “Account Executive” → discrepancy

Task 2: Train a Binary Classification Model

Train a model to predict whether a pair should be marked **verified** or **discrepancy**.

You may engineer features however you prefer (similarity metrics, text embeddings, keyword detection, etc.) and choose any reasonable model type.

Please:

- Describe your feature and model choices briefly.
- Report key metrics such as **Accuracy**, **Precision**, **Recall**, **F1**, and any other relevant evaluation measures (e.g., ROC-AUC or confusion matrix) that help demonstrate model performance.
- Provide any short observations on performance.

Deliverable: Notebook with the code, short summary, plots charts, whatever you think its useful

Task 3: Analyze Model Errors

Use your trained model to identify and analyze misclassifications. Look for patterns or clusters among the mistakes and briefly describe what you find.

If you wish, propose a simple way to detect potential issues automatically (e.g., confidence thresholds, drift detection, or unsupervised anomaly detection). Think about how a production system might catch problems in real time: for instance, if your model achieves 97% accuracy, how would you find and diagnose the remaining 3% of problematic cases? You can experiment with clustering, outlier detection, or other unsupervised methods to surface recurring patterns or new types of mismatches. This part is open-ended and meant to show your reasoning and creativity.

Deliverable: Notebook with the code, short summary, plots charts, whatever you think its useful

Submission

We understand you may be limited in time, so the goal is not perfection but thoughtful execution. Note any areas for improvement and describe what you would refine with more time or data. This task also evaluates your ability to manage time effectively - balance ambition across parts so each step is completed meaningfully. Treat this exercise as a single, continuous project rather than isolated tasks.

Focus on clearly documenting your approach, noting observations and possible improvements rather than trying to build a perfect solution.

You can either submit a ZIP file containing the CSV and notebook, or provide a link to a public GitHub repository.

Please submit:

1. `dataset.csv`
2. A notebook containing code and explanations of your approach, feature choices, model, metrics, and insights (inline is fine).
    - This task is open-ended: think through each step and how it flows into the next, and write down observations and clear areas for improvement.

Recombo.AI

Example dataset.csv

```
None
case_id, applicant_job_title, official_record_job_title, status, rule
C0001, senior software engineer, software engineer, discrepancy, Rule 4 (Hierarchy / seniority
difference)
C0002, receptionist at murphy oil, receptionist, verified, Rule 2 (Normalization)
C0003, n/a, plumber, verified, Rule 1 (Missing Title)
C0004, data scientist, associate, verified, Rule 5 (Departmental / generic vs specific)
C0005, software engineer, account executive, discrepancy, Rule 7 (Different job family)
C0006, full time, software engineer, verified, Rule 6 (Invalid title)
C0007, jr. data analyst, data analyst, discrepancy, Rule 4 (Hierarchy / seniority difference)
C0008, product manager, product manager, verified, Rule 3 (Exact / near match)
C0009, qa analyst, quality assurance analyst, verified, Rule 3 (Exact / near match)
C0010, sales, sales executive, verified, Rule 5 (Departmental / generic vs specific)
C0011, human resources, hr manager, verified, Rule 5 (Departmental / generic vs specific)
C0012, senior qa engineer, qa engineer ii, discrepancy, Rule 4 (Hierarchy / seniority difference)
C0013, software developer iii, software developer i, discrepancy, Rule 4 (Hierarchy / seniority
difference)
C0014, operations, operations manager, verified, Rule 5 (Departmental / generic vs specific)
C0015, executive vice president, vice president, discrepancy, Rule 4 (Hierarchy / seniority
difference)
C0016, abc123, mechanical engineer, verified, Rule 6 (Invalid title)
C0017, lead data scientist, principal data scientist, discrepancy, Rule 4 (Hierarchy / seniority
difference)
C0018, customer service representative, associate, verified, Rule 5 (Departmental / generic vs
specific)
C0019, sr. hr generalist, hr generalist, discrepancy, Rule 4 (Hierarchy / seniority difference)
C0020, system administrator, sys admin, verified, Rule 3 (Exact / near match)
C0021, seasonal, truck driver, verified, Rule 6 (Invalid title)
C0022, sales associate, marketing specialist, discrepancy, Rule 7 (Different job family)
C0023, software engineer intern, software engineer, discrepancy, Rule 4 (Hierarchy / seniority
difference)
C0024, assistant manager, manager, discrepancy, Rule 4 (Hierarchy / seniority difference)
C0025, software engineer, software engineer, verified, Rule 3 (Exact / near match)
C0026, receptionist, receptionist at hilton, verified, Rule 2 (Normalization)
C0027, office admin, administrator, verified, Rule 3 (Exact / near match)
C0028, plumber, master plumber, discrepancy, Rule 4 (Hierarchy / seniority difference)
C0029, cashier, sales representative, discrepancy, Rule 7 (Different job family)
C0030, student, trainee, verified, Rule 5 (Departmental / generic vs specific)
```