Predicting *E. coli* Antibiotic Resistance from Routine Lab Test Data

Author: Karim Makki
Course: 2-INF-150 Machine Learning (Fall 2025)

Abstract

The project applies supervised machine learning to predict antibiotic resistance for E. coli using a large, real-world lab testing dataset from Kaggle (195,795 rows). The raw data includes missing values and inconsistent labeling. and is organized as one row per test. I transform the data into a sample-level dataset (one row per Genome ID) and study two tasks: (A) predicting resistance outcomes for selected antibiotics using sample metadata, and (B) predict missing test outcomes using known results for other antibiotics. Logistic regression, random forests and histogram-based gradient boosting are compared across two random seeds. Results show that metadata-only While models achieve moderate performance, using known antibiotic outcomes greatly improves performance on Task B. F1 and AUROC. The report features a trial–error–analysis cycle, deeper experiments, explainability insights, and concrete data examples.

1. Introduction

Antibiotic resistance is a critical clinical problem. In practice, labs often test a subset of antibiotics and results may be missing or inconsistent. This project focuses on predictive modeling of resistance outcomes for *E. coli* in a large public dataset. The machine learning goal is to learn patterns from historical lab test records and generalize to new samples, enabling:

- automatic prediction of resistance for selected antibiotics, and
- inference of missing results when only part of a test panel is available.

This dataset is messy and incomplete by design, making it a realistic setting for evaluating data cleaning and modeling choices.

2. Problem formulation and task definition

The raw dataset contains one row per antibiotic test, whereas machine learning requires one row per sample. I transform the data into a sample level table where each genome ID is a training example and each antibiotic is one possible label. Since not all tests are conducted for all samples, the label matrix is sparse (many missing entries). This naturally leads to two related supervised learning problems: predicting resistance from metadata alone, and imputation of missing test results from partially observed panels.

Two tasks are studied:

- **Task A (metadata-only prediction):** predict resistance for selected antibiotics from sample metadata.
- **Task B (missing-outcome prediction):** predict a missing antibiotic result using metadata plus already-known results for other antibiotics.

Input vs output (with data interpretation)

Task A input (X): lab metadata that would be available at prediction time. These fields describe **how and under which standard the test was performed** (lab method, standard, platform, vendor), plus summary counts (how many tests were done for the sample). They are not resistance labels themselves, but contextual information about the sample and the testing process.

- Laboratory Typing Method
- Testing Standard
- Evidence
- Laboratory Typing Platform
- Vendor
- Testing Standard Year (median)
- Number of tests for the sample
- Number of unique antibiotics tested
- Has PubMed reference (binary)

- Has Source field (binary)

Task A output (Y): a binary resistance label for each selected antibiotic: 1 = resistant, 0 = not resistant. This is derived from the `Resistant Phenotype` column after label mapping.

Task B input (X): all Task A metadata features **plus** already-known antibiotic outcomes for the same sample (other antibiotics). This reflects the practical setting where a lab has results for some antibiotics and wants to infer the rest.

Task B output (Y): the missing resistance label for the target antibiotic.

This is standard supervised learning: the model learns a function f(X) -> Y from labeled samples and is evaluated on held-out data.

3. Dataset

Source: Kaggle E. coli Resistance Dataset (`archive/BVBRC_E.coli_Dataset.csv`).

Dataset statistics:

- Rows (one test per row): 195,795
- Unique samples (Genome ID): 12,780
- Unique antibiotics: 92
- Labeled rows (Resistant Phenotype present): 95,178

3.1 Raw data example

The raw data is **test-level**, so each row corresponds to a single antibiotic test for a sample. The table shows typical issues: missing resistance labels, missing metadata (method/standard), and inconsistent completeness across rows. This motivates the need for cleaning and aggregation.
For layout, column names are shortened and antibiotic names are abbreviated; full names appear in the dataset.

Genome ID	Antibiotic	Phenotype	Lab Method	Standard	Year
562.144628	CRO		MIC	CLSI	2018
562.5691	TMP/SMX		Broth dilution	CLSI	
562.144245	MEM	Susceptible	Broth dilution	EUCAST	2016
562.100003	TMP/SMX	Susceptible	Disk diffusion	EUCAST	
562.65814	FEP	Resistant			
562.143892	PIP/TAZ		MIC	CLSI	2021

3.2 Label mapping example

The `Resistant Phenotype` field contains multiple textual categories. To create a consistent binary label, I map resistant-like categories to 1 and susceptible-like categories to 0. This reduces noise and makes the prediction task well-defined.

raw_label	binary_label
Intermediate	0
Nonsusceptible	1
Resistant	1
Susceptible	0
Susceptible-dose dependent	0

3.3 Sample-level format

The raw test-level dataset is transformed to **one row per Genome ID**. Each antibiotic becomes an output column, while metadata is aggregated at the sample level. Missing outputs indicate tests that were not performed

for that sample.
The table below shows a **compact subset** of columns for readability; the full preview is in `reports/examples/sample_level_pre`

Genome ID	Lab Method	Standard	Evidence	Year	num_tests	num_abx	has_pubmed	y_AMP	y_
1045010.61	Broth dilution	CLSI			13	13	0	1.0	0.
1045010.62	Broth dilution	CLSI			13	13	0	1.0	0.
1045010.63	Broth dilution	CLSI			13	13	0	0.0	0.
1045010.64	Broth dilution	CLSI			13	13	0	1.0	0.
1328432.3	Broth dilution		Laboratory Method		16	16	1	0.0	0.

3.4 Missingness snapshot

Missingness is substantial and uneven across antibiotics. This is important for model design: complete-case multioutput training becomes impractical, while missing-outcome prediction becomes a realistic and valuable task. Antibiotic names are abbreviated for layout: AMP (ampicillin), CIP (ciprofloxacin), GEN (gentamicin), TMP/SMX, AMC, CXM, PIP/TAZ, CTX, CAZ.

Antibiotic	percent_missing
TMP/SMX	52.72
AMC	49.38
CXM	47.32
PIP/TAZ	26.02
AMP	25.27
CTX	24.63
CAZ	17.34
CIP	14.73
GEN	13.28

3.5 Selected antibiotics

The dataset contains 92 antibiotics, but many are too rare for reliable modeling. I therefore select a subset with **enough labeled data and class balance** so that both resistant and non-resistant examples exist.

Selection rules (applied to labeled rows):

- At least 2000 labeled samples per antibiotic
- At least 100 resistant and 100 non-resistant cases
- Keep the top 9 antibiotics by coverage after filtering

Final selected antibiotics (used in all experiments):

- gentamicin
- ciprofloxacin
- ceftazidime
- ampicillin
- cefotaxime
- piperacillin/tazobactam
- cefuroxime
- amoxicillin/clavulanic acid
- trimethoprim/sulfamethoxazole

4. Data preparation

The preparation pipeline turns a noisy, test-level table into an ML-ready dataset while keeping the process transparent and reproducible.

1. **Label mapping**
   Convert the text `Resistant Phenotype` field into a binary label using the mapping in Section 3.2. Rows with unknown or missing labels remain unlabeled.
2. **Sample-level aggregation**
   - Group by `Genome ID` so each sample becomes a training example.
   - Pivot antibiotic labels into a multi-label output vector (one column per antibiotic).
   - Preserve missing labels as NaN to reflect tests that were not performed.
3. **Feature engineering**
   - Categorical metadata: lab method, testing standard, evidence, platform, vendor.
   - Numeric metadata: testing year (median), count of tests, number of unique antibiotics.
   - Binary metadata: `has_pubmed` and `has_source` indicators.
     These features represent the lab context rather than resistance itself, which helps avoid leakage.
   - Categorical features are later one-hot encoded during modeling, and numeric features are standardized.
4. **Missingness handling**
   - Metadata is imputed (most-frequent for categorical, median for numeric).
   - Antibiotic outcomes are not imputed by default; missingness is explicitly modeled in Task B, and an imputed variant is tested for multioutput training.
5. **Train/test split**
   Perform an 80/20 split at the sample level with fixed random seeds to make experiments repeatable.

5. Methods

Models

Three model families are used to balance interpretability, robustness, and performance on tabular data:

- **Logistic Regression (logreg):** a strong linear baseline with class weighting. It is easy to interpret and serves as a reference point for more complex models.
- **Random Forest (rf):** a non-linear ensemble that captures feature interactions without heavy preprocessing. It is robust to noisy features and can handle mixed data types after encoding.
- **Histogram Gradient Boosting (hgb):** a modern boosting model optimized for tabular data. It often performs well with complex, non-linear patterns and is efficient on medium-sized datasets.

All models are trained with class weighting to reduce bias toward the majority (susceptible) class. Categorical features are one-hot encoded, numeric features are standardized, and missing metadata is imputed as described in Section 4.

Tasks and pipelines

Multiple pipelines are compared to isolate the effect of **task formulation**:

- **Task A independent:** train one model per antibiotic using metadata only. This maximizes usable data per antibiotic and is a strong baseline.
- **Task A multioutput complete-case:** train a multi-output model using only samples with all labels present. This tests the ideal multi-label formulation but suffers from severe data loss due to missingness.
- **Task A multioutput impute:** allow multi-output training by imputing missing antibiotic outcomes (treated as unknown). This tests whether multi-label structure helps when missingness is handled.
- **Task B missing outcomes:** train per-antibiotic models using metadata plus known outcomes for other antibiotics. This directly targets the practical problem of inferring missing test results.

Evaluation

- **Metrics:** accuracy, F1-score, and AUROC. F1 and AUROC are emphasized because resistance is imbalanced for many antibiotics.
- **Seeds:** experiments are run with seeds 42 and 1337 to check stability.
- **Split:** 80/20 train/test split at the sample level to avoid leakage across samples.
- **Aggregation:** macro-averages across antibiotics summarize overall performance for each task/model.

6. Results

6.1 Summary by task and model (seed-averaged)

Table 1 reports macro-averaged performance (across antibiotics) for each task and model, averaged over two random seeds. This highlights how **task formulation** changes performance and provides a fair comparison across model families.

task	model	accuracy	f1	roc_auc
task_a_independent	hgb	0.834	0.504	0.781
task_a_independent	logreg	0.728	0.486	0.775
task_a_independent	rf	0.762	0.515	0.781
task_a_multioutput_complete	hgb	0.704	0.106	0.5
task_a_multioutput_complete	logreg	0.741	0.326	0.7
task_a_multioutput_complete	rf	0.741	0.326	0.709
task_a_multioutput_impute	hgb	0.893	0.54	0.869
task_a_multioutput_impute	logreg	0.762	0.485	0.841
task_a_multioutput_impute	rf	0.752	0.491	0.869
task_b_missing_outcomes	hgb	0.893	0.745	0.919
task_b_missing_outcomes	logreg	0.855	0.704	0.908
task_b_missing_outcomes	rf	0.87	0.725	0.918

6.2 Best model per task (by F1)

Table 2 summarizes the **single best model** for each task based on F1. This makes it easier to identify the most effective pipeline without over-emphasizing small differences between similar models.

task	model	accuracy	f1	roc_auc
task_a_independent	rf	0.762	0.515	0.781
task_a_multioutput_complete	logreg	0.741	0.326	0.7
task_a_multioutput_impute	hgb	0.893	0.54	0.869
task_b_missing_outcomes	hgb	0.893	0.745	0.919

Overall, HGB is the strongest choice for tasks that can benefit from non-linear interactions and larger training sets (impute and Task B). For metadata-only prediction, RF and HGB are close, while the complete-case multioutput setting remains limited by data sparsity regardless of model choice.

6.3 Key observations

- **Task B is consistently best because it adds clinically meaningful signal.**
  When known antibiotic outcomes are included as inputs, models can learn co-resistance patterns. This yields a large jump in F1 (roughly 0.50 -> 0.74–0.76) and AUROC (~0.78 -> ~0.92). The gain is consistent across models and random seeds, indicating a robust effect rather than noise.
- **Metadata-only prediction is feasible but limited.**
  Task A (independent) uses only metadata such as testing standard and lab method. This achieves moderate performance (F1 ~0.50), which confirms there is signal in metadata, but it is not sufficient for high-confidence prediction on its own.
- **Complete-case multioutput learning is unstable due to missingness.**
  Many antibiotics have >40–50% missing labels. Restricting to complete-case samples removes most data, producing unstable results and low F1. This is an expected failure mode and highlights the importance of handling missing outcomes explicitly.
- **Non-linear models offer consistent gains.**
  Histogram Gradient Boosting (HGB) provides the strongest macro F1 and AUROC across tasks, suggesting that non-linear interactions between metadata features and resistance outcomes exist. Random Forests are competitive, while Logistic Regression is a strong but weaker baseline.

- **Performance varies by antibiotic, and class imbalance matters.**
  Antibiotics with low resistance rates (e.g., piperacillin/tazobactam) show lower F1 despite decent accuracy, which indicates that accuracy alone is misleading under imbalance. F1 and AUROC are more informative in this setting.

These observations connect directly to the figures in `reports/figures/`, which visualize coverage, missingness, and per-antibiotic performance.

7. Discussion

The results indicate that **task formulation is more important than the model family** for this dataset. The complete-case multioutput setup is theoretically appealing but practically fragile because the data is extremely sparse; only a tiny subset of samples has full labels, which leads to unstable metrics and poor generalization. When the task is reframed as independent per-antibiotic prediction (Task A), performance becomes more stable, confirming that the dataset carries usable signal in metadata. The strongest improvement comes from Task B, where known outcomes for other antibiotics are included as inputs; this aligns with clinical intuition that resistance patterns co-occur across antibiotics.

Model comparisons suggest **non-linear models are consistently better** than linear baselines, likely because resistance depends on interacting factors (e.g., lab method, testing standard, and antibiotic type). However, performance is still uneven across antibiotics, driven by missingness and class imbalance. This is why F1 and AUROC are emphasized over accuracy: they are more informative when resistance is rare. Overall, the findings support the practical use-case of predicting missing test outcomes rather than attempting to predict all antibiotics solely from metadata.

7.1 Trial–error–analysis cycle

To satisfy the required trial–error–analysis cycle, I iterated through multiple formulations and documented the outcome of each change.

Cycle 1: Complete-case multioutput

- **Attempt:** Train a multi-output classifier using only samples with all antibiotic labels present.
- **Observation:** Unstable results and low F1 for some seeds.
- **Analysis:** Complete-case filtering leaves only a tiny subset of samples, so the model overfits and fails to generalize.
- **Action:** Move away from complete-case training.

Cycle 2: Multioutput with imputation

- **Attempt:** Keep a multi-output formulation but impute missing antibiotic outcomes so more samples can be used.
- **Observation:** F1 improves compared to complete-case, but performance is still below the strongest baselines.
- **Analysis:** Imputation introduces noise because unknown labels are treated as if they were observed. The model sees mixed-quality targets.
- **Action:** Switch to independent per-antibiotic models that avoid label imputation.

Cycle 3: Independent per-antibiotic models

- **Attempt:** Train one model per antibiotic using metadata only (Task A independent).
- **Observation:** Stable, moderate performance (F1 around 0.50).
- **Analysis:** Metadata contains signal, but it is not sufficient to capture the full resistance patterns.
- **Action:** Add known antibiotic outcomes as additional inputs.

Cycle 4: Missing-outcome prediction (Task B)

- **Attempt:** Predict a target antibiotic using metadata plus known outcomes for other antibiotics.
- **Observation:** Best overall performance (F1 ~0.74–0.76).
- **Analysis:** Resistance outcomes co-occur across antibiotics; using observed outcomes supplies crucial predictive signal.
- **Result:** Task B becomes the primary recommended formulation.

7.2 Deeper experiment

Research question: Does incorporating known antibiotic outcomes improve prediction beyond metadata alone?

Experimental design:

- **Control:** Task A independent (metadata only).
- **Treatment:** Task B (metadata + known outcomes for other antibiotics).
- **Models:** logreg, rf, hgb (same train/test split and seeds).
- **Metrics:** macro F1 and AUROC across antibiotics.

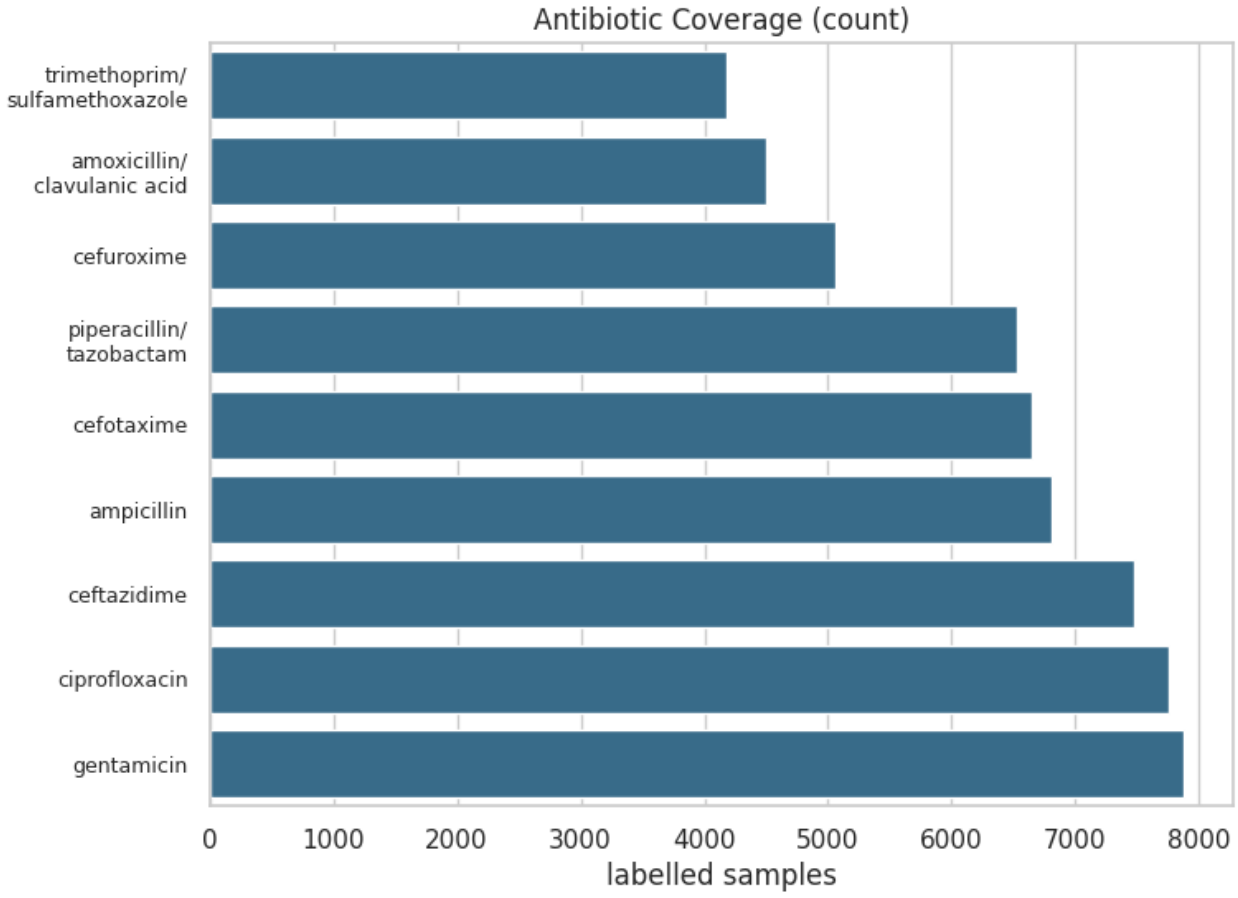
Results:
Task B consistently outperforms the metadata-only baseline. F1 increases from roughly 0.50 to 0.74–0.76, and AUROC increases from around 0.78 to ~0.92. These gains are consistent across models and seeds.

Interpretation:
Antibiotic resistance is not independent across tests; outcomes for one antibiotic provide meaningful context for others. This supports the practical clinical use-case of filling in missing tests based on observed panels.

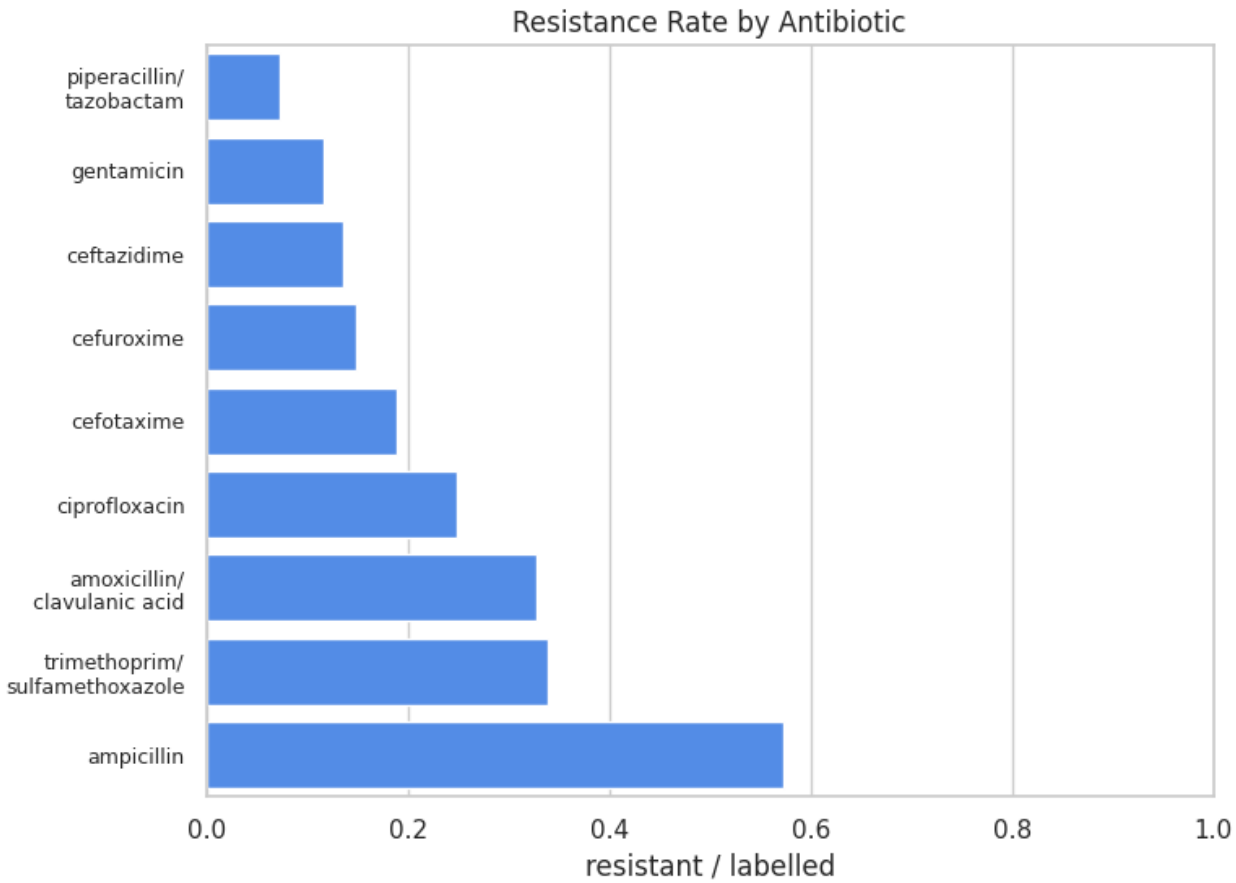
8. Visualizations

The script `python3 scripts/visualize_results.py` generates the figures below in `reports/figures/`. Each plot highlights a different aspect of the dataset or model behavior.

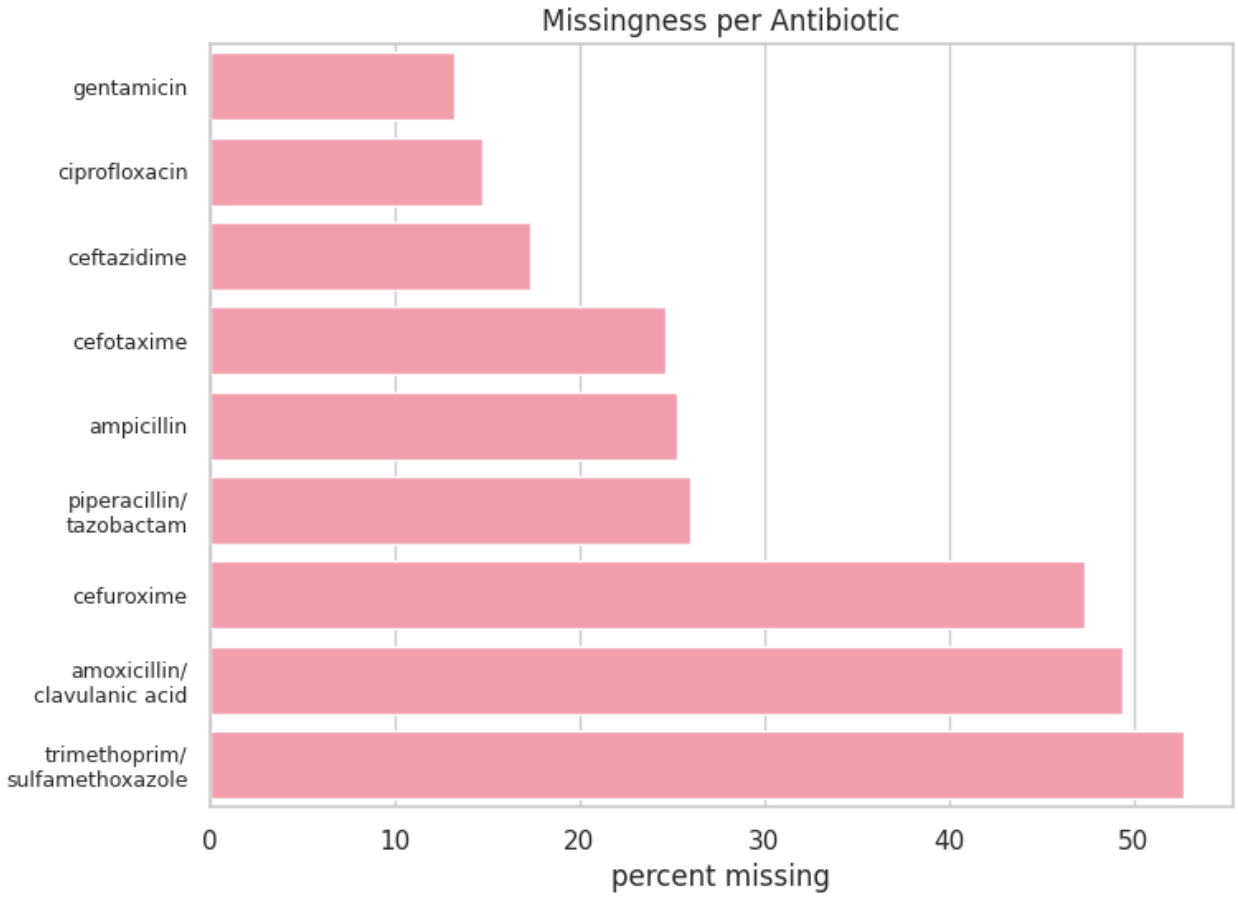
- `antibiotic_counts.png`: number of labeled samples per antibiotic.
  What it shows: coverage differences across antibiotics and why some labels are more reliable than others.



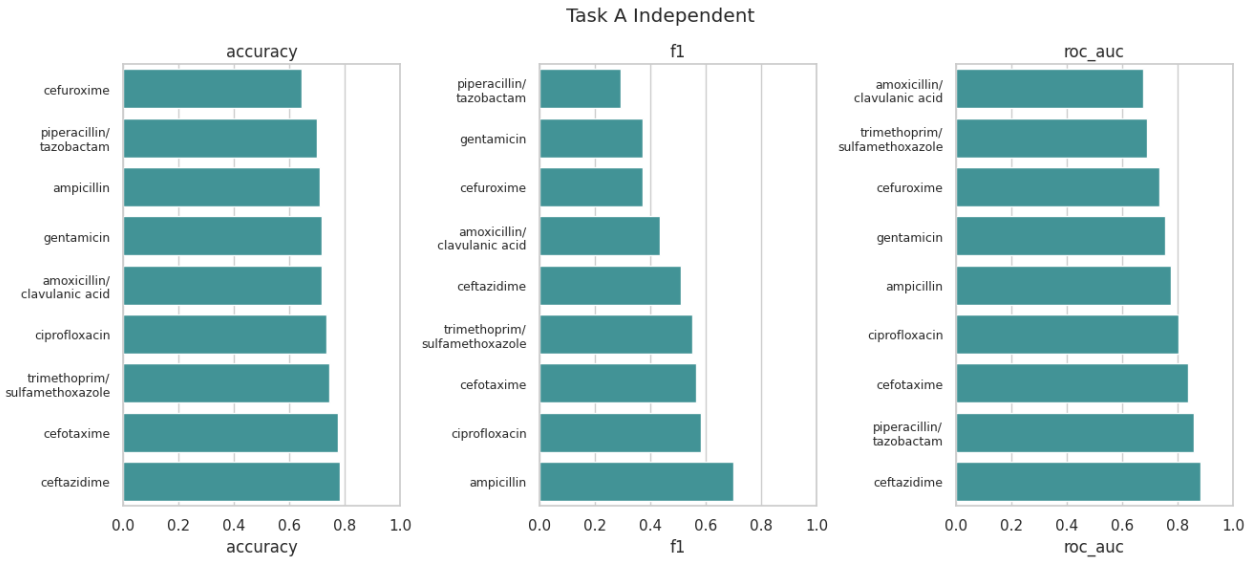
- `antibiotic_resistance_rate.png`: resistance rate per antibiotic.
  What it shows: class imbalance; antibiotics with low resistance are harder to model using F1.



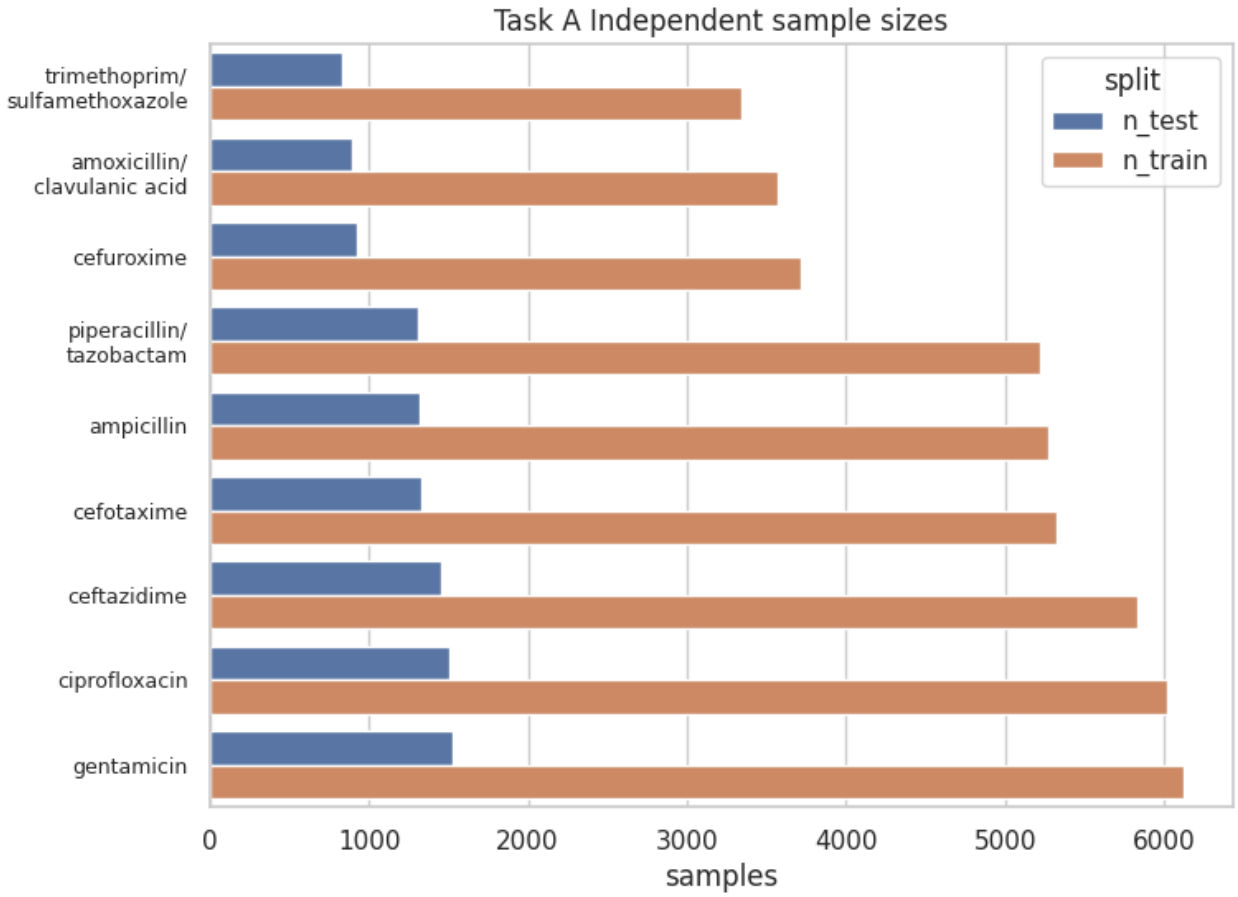
- `missingness_by_antibiotic.png`: percent missing labels per antibiotic.
  What it shows: why complete-case multioutput training is impractical.



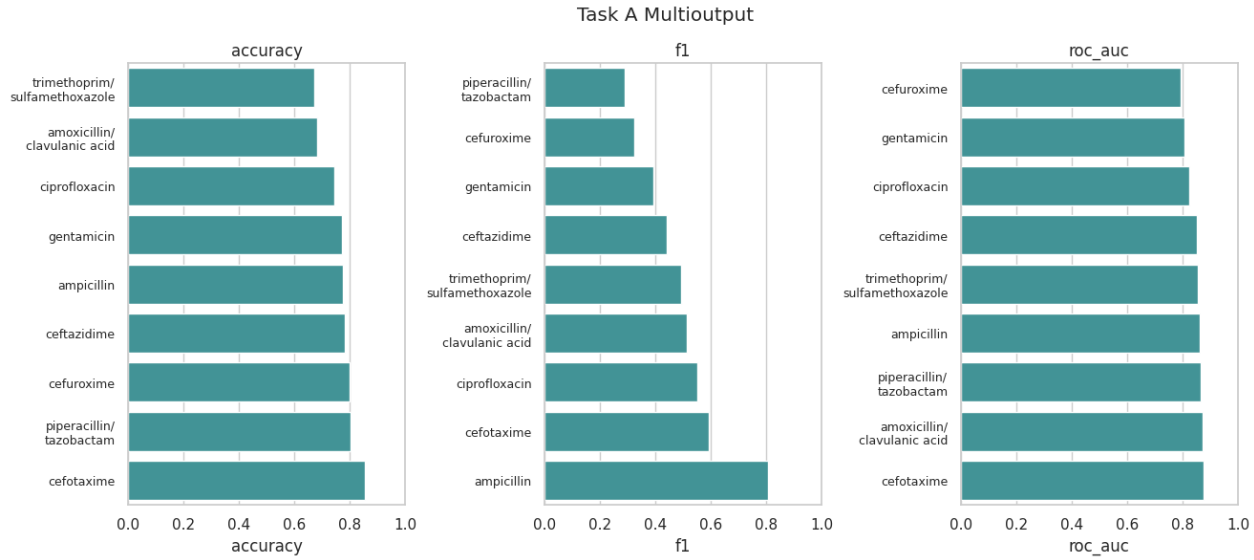
- `task_a_independent_metrics.png`: accuracy/F1/AUROC per antibiotic for Task A.
  What it shows: which antibiotics are predictable using metadata alone.



- `task_a_independent_sample_sizes.png`: training/test sizes per antibiotic for Task A.
  What it shows: how label availability differs across antibiotics and affects results.

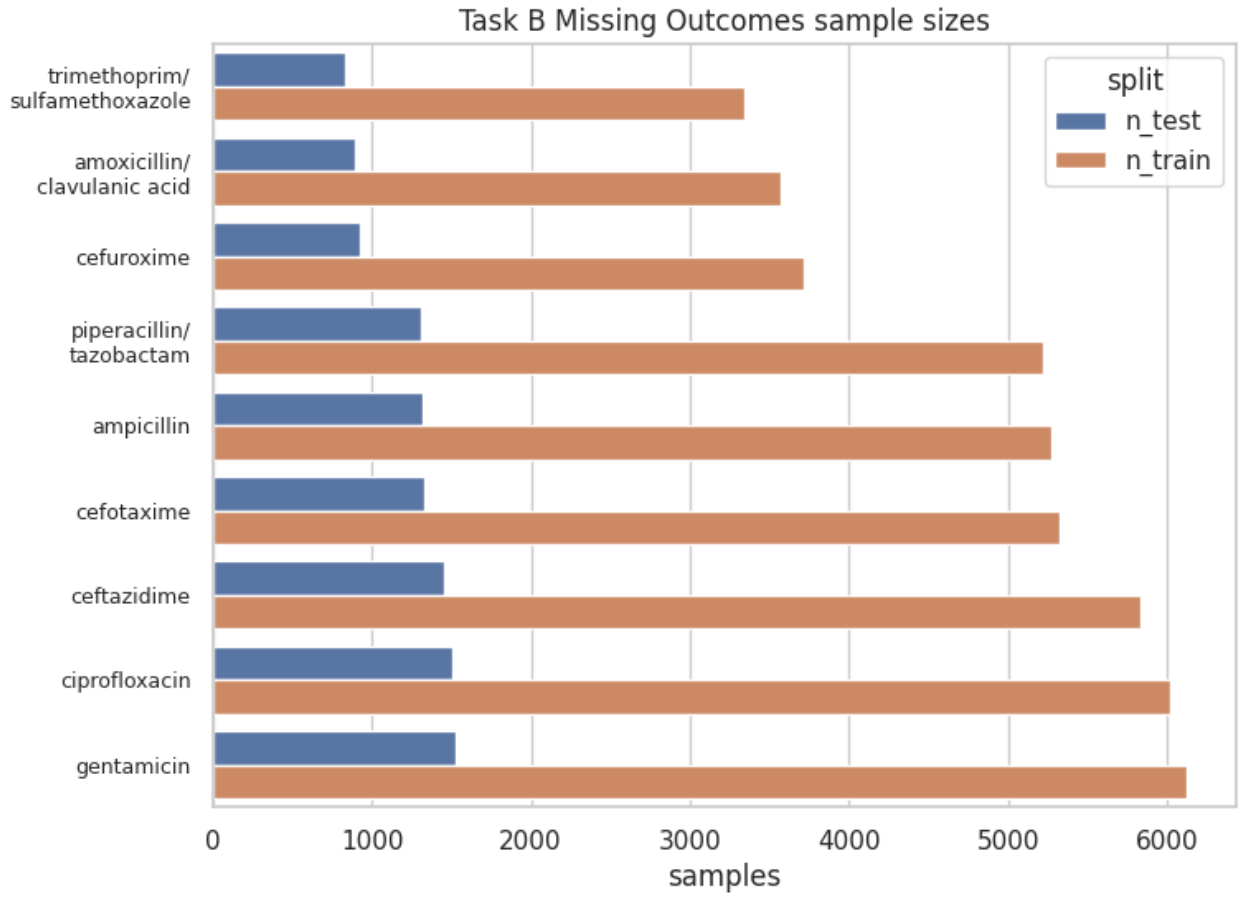


- `task_a_multioutput_metrics.png`: per-antibiotic accuracy/F1/AUROC for the multioutput variant. **What it shows:** which antibiotics benefit (or not) from the multi-label formulation.

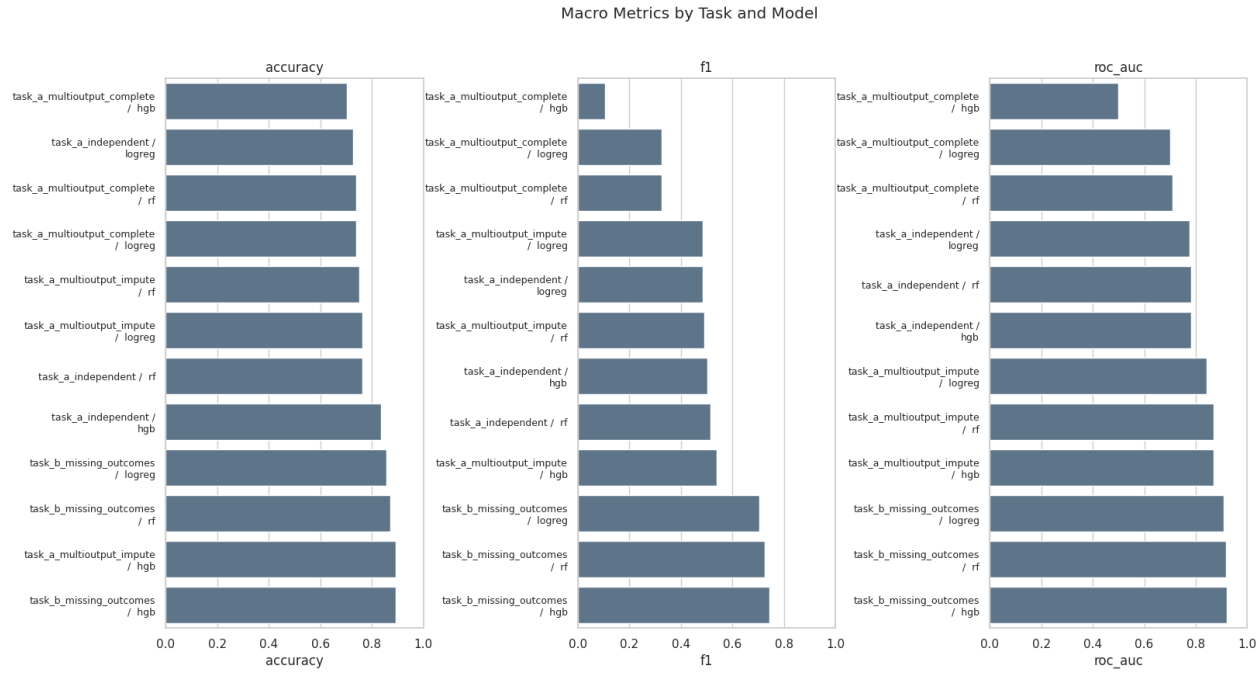


- `task_b_missing_outcomes_metrics.png`: accuracy/F1/AUROC per antibiotic for Task B. **What it shows:** the improvement when known outcomes are included.

- `task_b_missing_outcomes_sample_sizes.png`: training/test sizes per antibiotic for Task B.
  What it shows: how many samples are available once partial panels are used.



- `task_model_macro_metrics.png`: macro accuracy/F1/AUROC aggregated by task and model.
  What it shows: a compact comparison of task formulations and model families using the same metrics as Table 1.



9. Limitations

- Only metadata features are used; genomic or clinical features that may strongly influence resistance are not included.
- Train/test split is random; a temporal split or site-based split could better simulate deployment conditions.
- Multioutput complete-case learning is limited by missingness, which reduces training data to an impractically small set.
- Labels are noisy and derived from heterogeneous lab standards, which introduces uncertainty that is not explicitly modeled.

10. Future work

- Add calibrated probabilities and threshold tuning per antibiotic for better decision-making.
- Explore semi-supervised learning to exploit the large number of unlabeled tests.
- Include feature importance or SHAP analysis to interpret which metadata drives predictions.
- Evaluate time-based splits to test robustness to shifts in testing standards over time.
- Incorporate additional data sources (genomic features or patient-level metadata) if available.

11. Project positioning vs assignment ideas

This project fits most strongly into the following suggested categories from the assignment:

- **Applying ML techniques to a very atypical data set:** the dataset is noisy, incomplete, and not ML-ready (one row per test rather than per sample).
- **Testing effects of preprocessing techniques or different algorithms:** I compare multiple model families (logreg, RF, HGB) and multiple task formulations (complete-case, impute, independent, missing-outcomes).
- **Explaining or finding atypical model behavior:** the complete-case multioutput model fails due to missingness, which is analyzed and resolved by changing the task formulation.

It does **not** include unsupervised visualization or semi-supervised methods yet, which are possible extensions.

12. Conclusion

This project demonstrates a full ML pipeline on a real, messy biomedical dataset and shows how careful task formulation can overcome practical data limitations. By converting test-level data into sample-level features, it

becomes possible to model antibiotic resistance in a reproducible way. The strongest and most consistent result is that **using known antibiotic outcomes (Task B) yields substantially better performance** than relying on metadata alone.

The report documents a realistic development cycle: an initial multioutput approach fails due to missingness, an imputed variant improves but remains limited, and a reformulated task (missing-outcome prediction) delivers the best results. This emphasizes that the central challenge is not only model selection but also how the prediction problem is framed.

Overall, the project provides a solid baseline for resistance prediction with clear limitations and a roadmap for future improvements. It meets the assignment goal of applying ML to a complex dataset, analyzing failures, and improving the approach through targeted experiments.