



# Hackathon Judging Criteria

Your project will be evaluated in **2 main areas**:

- **Model Performance (60%)**
  - **Practical Considerations (40%)**
- 

**1**

## Primary Metrics (60%)

These measures how well your model predicts who will get sick.



### F2-Score (Most Important)

- Focuses more on **recall than precision**
- Catching sick people is more important than avoiding false alarms
- Missing a sick person = worse than flagging a healthy one



Your model should prioritize identifying at-risk individuals.

---



### PR-AUC (Precision-Recall AUC)

- Best for **imbalanced datasets**
- Most people are healthy → very few become sick
- Measures performance when positive cases are rare



Your model must work well even when disease cases are limited.

---



### ROC-AUC

- Industry standard metric
  - Helps compare your model with published research
  - Measures overall classification ability
- 

**2**

## Additional Criteria (40%)

These focus on real-world practicality and ethics.

---



### No Data Leakage

You cannot use features that already reveal the disease.

Example:

- If someone is taking diabetes medication → they already have diabetes
- Using that feature = cheating



Judges will audit your feature set.

---



### Real-World Usability

Your model must handle:

- Noisy inputs
- Missing values
- Inconsistent self-reported data

Remember:

You're not working with perfect hospital records — only self-reported data.

---

## Cost Efficiency

- Simple > Complex
- Lightweight models > Heavy API-based systems
- Fast runtime is better

Avoid overengineering.

---

## Open Source Only

- All tools must be open-source
  - Code must be reproducible
  - No proprietary black-box APIs
- 

## Explainability

You must explain:

- Why was this person flagged?
- Which features contributed most?

Both:

- Users
- Doctors

should understand the reasoning.

---

## Fairness

Your model must NOT discriminate based on:

- Age

- Gender
- Ethnicity

Judges will check for bias.

---

## Bonus Points

Extra credit for:

- ◆ Creative feature engineering
  - ◆ Discovering non-obvious correlations
  - ◆ Clean, production-ready code
  - ◆ Clear documentation
- 

## What This Means Strategically

To win:

1. Maximize **F2-score**
2. Avoid leakage
3. Keep model simple & explainable
4. Handle missing/noisy data well
5. Show fairness checks
6. Keep everything open-source