ML model structure

- Data Collection
- Data Preprocessing
- Model(Algorithm) selection
- Model Training
- Evaluation
- Testing
- Deployment

ML algorithms

- Algorithm=Ketma ketlik
- O'qituvchi Metodi= Algorithm

ML algorithms family

- Classification
- Regression

ML Algorithm Types by Task

1. **Classification**

- Task: **Predict a category or class**.
- Output is **discrete**, e.g.,
  - Sick / Healthy
  - Yes / No
  - Dog / Cat / Bird

2. **Regression**

- Task: **Predict a continuous number**.
- Output is **numeric**, e.g.,
  - Cow's milk production → 12.5 liters
  - House price → $120,000
  - Temperature → 37.8°C

✅ **In short:**

- Classification = **classes/categories**
- Regression = **numbers/continuous values**

ML algorithms family

1️⃣ Linear Models

- **Linear Regression** → predicts a **continuous number**.
  - Example: Predict milk yield in cows.
- **Logistic Regression** → predicts a **category/class** (like Yes/No).
  - Example: Predict if a cow has mastitis.

Difference: Linear → continuous output; Logistic → categorical output.

2️⃣ Tree-Based Models

- **Decision Tree** → a single tree, easy to visualize, may **overfit** small datasets.
- **Random Forest** → a **collection of many trees** (ensemble), more robust, reduces overfitting.

Difference: Decision Tree = one tree; Random Forest = many trees combined.

3️⃣ Distance-Based Models

- **KNN (K-Nearest Neighbors)** → predicts based on the **closest points** in the dataset.
- **SVM (Support Vector Machine)** → finds the **best boundary/hyperplane** to separate classes.

Difference: KNN uses **distance to neighbors**; SVM uses **geometric boundary**.

4️⃣ Ensemble Models

- **Random Forest** → bagging approach (many trees, majority vote).
- **Gradient Boosting** → sequentially builds trees, **each fixes errors of the previous**.

Difference: Random Forest = parallel trees; Gradient Boosting = sequential trees improving gradually.

Linear

- Linear Regression
- Logistic Regression

Tree-Based

- Decision Tree
- Random Forest

Distance-Based

- KNN
- SVM

Ensemble

- Random Forest
- Gradient Boosting

Here's a clear separation of the algorithms you listed into **Classifier** and **Regression**:

Classifier (predict categories / classes)

- **Linear**: Logistic Regression
- **Tree-Based**: Decision Tree, Random Forest
- **Distance-Based**: KNN, SVM (SVC)
- **Ensemble**: Random Forest, Gradient Boosting

Regression (predict continuous numbers)

- **Linear**: Linear Regression
- **Tree-Based**: Decision Tree, Random Forest
- **Distance-Based**: KNN, SVM (SVR)

- **Ensemble**: Random Forest, Gradient Boosting

✅ **Notes:**

1. Some algorithms like **Decision Tree, Random Forest, KNN, Gradient Boosting, SVM** can handle **both classification and regression**, depending on how they are used.
2. Logistic Regression → classification only.
3. Linear Regression → regression only.

Logistic Regression

- Logistic Regression
- Classification
- **REGRESSION UCHUN ISHLAMAYDI**

Logistic Regression

- Classification tasklar uchun effiktive (**asosan binary uchun**) algorithmlardan biri:
    - **Sodda**
    - **Kichik datasetlarda yaxshi ishlaydi**

Logistic Regression

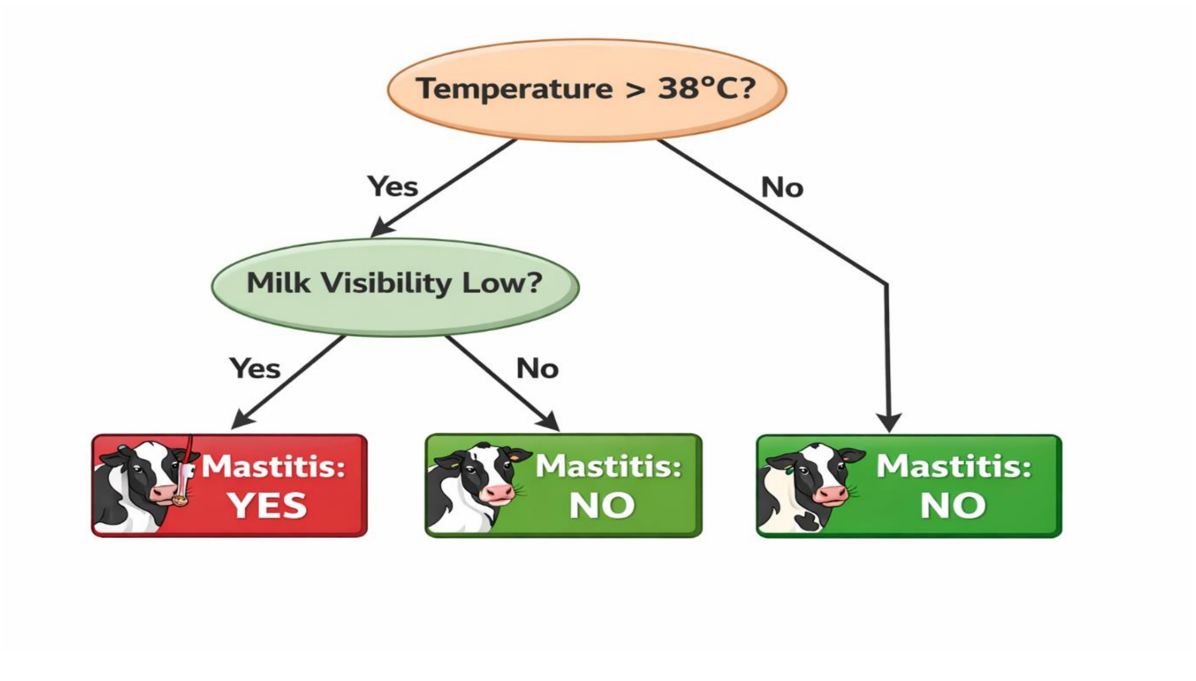
- Bosqichlari:
    - Data Preprocessing
    - x,y
    - Splitting(train, test)
    - Training(fitting)
    - Predicting
    - Evaluating

Biz model tanlamaymiz, balki model yaratamiz. Model tayyor bo‘lib, faqat foydalanishga mo‘ljallangan dasturdir.

Algorithm esa retsept yoki ketma-ketlikdir — u tayyor emas va uni ishlatishdan oldin qo‘llash tartibini belgilashimiz kerak. Algorithm tanlaymiz, chunki u model yaratishda foydalanadigan ketma-ketlikni belgilaydi.

Biz yaratmoqchi bo‘lgan loyiha — AI model deyiladi. Qaysi algorithm yaxshi bo‘lishi esa datasetimizga bog‘liq. Datasetimiz bizning o‘quvchilarimizdir.

- Linear → simple lines.
- Tree → decision paths. (During training, the tree only builds paths that exist in the training data. It doesn’t create routes for impossible or unseen combinations of features. So the tree grows based on actual patterns in the dataset, not all theoretical combinations. Think of it like a map of the dataset: it only draws roads where data actually travels. At each node, the decision tree chooses the feature and value (or category) that best separates the classes**—in your example, **sick vs healthy cows.)
- Distance → based on closeness.
- Ensemble → many models combined for better results.



---

1 Linear Regression (predict numbers)

Goal: Find the best line that predicts (y) from (x).

Steps:

1. Start with a line equation: (y = m \cdot x + b)
2. Measure **how far off the line is** from actual points ((y_{\text{true}} - y_{\text{predicted}})) → called **error/residual**
3. Adjust (m) (slope) and (b) (intercept) to **minimize the total error** (usually using **least squares**)
4. Once the error is minimized, the line is "trained" → we can **predict new (y)** for any (x)

Intuition: The line "best fits" the trend of the data points.

2️⃣ Logistic Regression (predict classes)

Goal: Predict **probabilities** that something belongs to a class (0 or 1).

Steps:

1. Start with a linear equation like Linear Regression: (z = m \cdot x + b)
2. Pass (z) through a **sigmoid function**:

[
p = \frac{1}{1 + e^{-z}}
]

- Output (p) is between 0 and 1 → probability of class 1

3. Convert probability to class:
   - (p \ge 0.5) → Class 1
   - (p < 0.5) → Class 0
4. Training finds (m) and (b) that **maximize the likelihood** of correctly predicting the classes (using **log-loss / cross-entropy**).

Intuition: Logistic Regression draws a "line" but uses it to **separate classes probabilistically**, not predict exact numbers.

✅ Key difference:

- Linear Regression → predicts **actual numbers**
- Logistic Regression → predicts **probabilities**, then converts to **categories**
