Progress Report on App Development (Week 04)

-by Digvijay Jadhav

An Intern in UpSkill Campus, 10/02/2026

Digvijay Jadhav

I am pleased to present you with a comprehensive report on data science and machine learning (Week 04), which provides an overview of the process, challenges, and best practices for successful data science and machine learning. This report aims to make understanding of the key aspects of data science and machine learning and making informed decisions in this domain.

What is Machine Learning?

Machine Learning (ML) is when **computers learn patterns from data** and make decisions **without being explicitly programmed** for every rule.

Instead of:

"If this happens, do that"
ML says:
"Here's data—figure out the rule yourself."

Tiny real-life examples

- **Netflix / YouTube** → recommends videos you'll probably like
- **Email** → spam vs non-spam
- **Google Maps** → predicts traffic
- **Phone camera** → face detection
- **Banking apps** → fraud detection

Core types of Machine Learning

1. **Supervised Learning**
   Data is labeled
   □ Example: Predicting *house price* from past prices
2. **Unsupervised Learning**
   No labels, just patterns
   □ Example: Grouping customers by behavior
3. **Reinforcement Learning**
   Learn by reward & punishment
   □ Example: Game AI, robots

A tiny code taste (Python □)

```
from sklearn.linear_model import LinearRegression

X = [[1], [2], [3], [4]]   # input data
y = [2, 4, 6, 8]           # output data

model = LinearRegression()
model.fit(X, y)

print(model.predict([[5]]))  # predicts 10
```

Why probability matters in Data Science & ML

Real-world data is:

- noisy

- incomplete
- uncertain

Probability gives us a **mathematical way to handle uncertainty** and make *best-possible decisions*, not perfect ones.

ML models don't say "this is 100% true"
They say "this is **likely** true"

Key probability concepts you MUST know

1. Random Variable

A variable whose value depends on chance.

- Discrete → number of emails, clicks
- Continuous → height, temperature

Example:

```
X = number of customers visiting a website today
```

---

2. Probability Distribution

Describes how likely different outcomes are.

Common ones in ML:

Distribution	Where used
Bernoulli	Yes/No (spam or not)
Binomial	Number of successes
Normal (Gaussian)	Natural data, errors
Poisson	Event counts
Exponential	Time between events

---

3. Mean, Variance, Standard Deviation

They summarize data behavior.

- **Mean** → average
- **Variance** → spread
- **Std Dev** → uncertainty

In ML:

- Loss functions assume certain distributions
- Feature scaling depends on variance

---

4. Conditional Probability

Probability of A **given** B.

P(A|B)=P(A∩B)P(B)P(A|B) = \frac{P(A \cap B)}{P(B)}P(A|B)=P(B)P(A∩B)

Example:

Probability email is spam **given** it contains “FREE”

Used in:

- Naive Bayes
- Recommendation systems
- Risk prediction

---

5. Bayes’ Theorem (VERY IMPORTANT)

P(A|B)=P(B|A)P(A)P(B)P(A|B) = \frac{P(B|A)P(A)}{P(B)}P(A|B)=P(B)P(B|A)P(A)

Plain English:

Update your belief when new evidence arrives

Used in:

- Spam filtering
- Medical diagnosis
- Fraud detection
- Bayesian ML

---

6. Independence

Two events are independent if one doesn’t affect the other.

Naive Bayes **assumes independence** (even if it’s not perfectly true, it works shockingly well).

Probability inside Machine Learning Algorithms

Naive Bayes

Pure probability-based classifier.

Uses:

- Conditional probability

- Bayes theorem

Logistic Regression

Outputs:

Probability between 0 and 1

Uses:

- Bernoulli distribution
- Maximum Likelihood Estimation

Neural Networks

- Output layer = probability distribution (Softmax)
- Loss = negative log-likelihood

Decision Trees & Random Forest

- Splits based on probability of class purity
- Uses entropy (from probability)

Clustering (GMM)

- Uses probability to assign points to clusters

Probability vs Statistics (important distinction)

Probability	Statistics
Theory → data	Data → theory
Assumes model	Learns model
Used in ML design	Used in evaluation

ML = **Probability + Statistics + Optimization**

Simple ML-style probability example

```
P_spam = 0.3
P_free_given_spam = 0.8
P_free = 0.4

P_spam_given_free = (P_free_given_spam * P_spam) / P_free
```

```
print(P_spam_given_free)
```

What do we mean by "basic algorithms"?

They are **foundational methods** used to:

- analyze data
- find patterns
- make predictions
- support advanced ML models

Most real projects still rely on these.

---

1. Linear Regression

 Type: Supervised (Regression)

What it does:
Predicts a **continuous value** using a straight-line relationship.

Example:
House price prediction, salary prediction

Idea:

$y=mx+cy = mx + cy=mx+c$

Why important:

- Simplest predictive model
- Base for many advanced algorithms

---

2. Logistic Regression

 Type: Supervised (Classification)

What it does:
Predicts **probability of a class** (0 or 1).

Example:
Spam detection, disease prediction

Output:
Value between **0 and 1**

---

3. K-Nearest Neighbors (KNN)

 Type: Supervised

What it does:
Classifies data based on **nearest neighbors**.

Example:
Movie recommendation, pattern recognition

Key idea:

"Tell me who your neighbors are, I'll tell you who you are."

---

4. Naive Bayes

 Type: Supervised (Probabilistic)

What it does:
Uses **Bayes' theorem** to classify data.

Example:
Spam filtering, sentiment analysis

Why used:

- Fast
- Works well on text data

---

5. Decision Tree

 Type: Supervised

What it does:
Splits data using **if–else rules**.

Example:
Loan approval, student performance

Advantage:
Easy to understand & visualize

---

6. K-Means Clustering

 Type: Unsupervised

What it does:
Groups data into **K clusters**.

Example:
Customer segmentation, image compression

Key idea:
Minimize distance within clusters.

---

7. Principal Component Analysis (PCA)

 Type: Unsupervised (Dimensionality Reduction)

What it does:
Reduces features while keeping most information.

Example:
Data visualization, speeding up ML models

---

8. Support Vector Machine (SVM)

 Type: Supervised

What it does:
Finds the **best boundary** between classes.

Example:
Face recognition, bioinformatics

---

9. Random Forest (basic ensemble)

 Type: Supervised

What it does:
Combines many decision trees to improve accuracy.

Example:
Fraud detection, stock prediction

What is Density Estimation?

Density estimation is the process of **estimating the probability distribution** (probability density function, PDF) of a dataset **from data**.

In simple words:

"Given data samples, figure out **how likely** different values are."

Why density estimation matters in DS & ML

It helps to:

- model **uncertainty**
- detect **anomalies / outliers**
- generate **new data**
- perform **probabilistic classification**
- understand data shape (skewed, multimodal, etc.)

Types of Density Estimation

1. Parametric Density Estimation

Assumes data follows a known distribution (Normal, Poisson, etc.).

Example: Gaussian Distribution

p(x)=12πσ2e−(x−μ)22σ2p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}p(x)=2πσ21 e−2σ2(x−μ)2

Steps:

1. Assume distribution type
2. Estimate parameters (mean μ, variance σ^2)
3. Use the PDF

 Pros: Simple, fast
 Cons: Wrong assumption → wrong model

---

2. Non-Parametric Density Estimation

No assumption about data distribution.

(a) Histogram

- Divide data into bins
- Count frequencies

 Simple but **blocky** and bin-dependent.

(b) Kernel Density Estimation (KDE)

Smooth version of histogram.

p^(x)=1nh∑i=1nK(x−xih)\hat{p}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)p^(x)=nh1i=1∑n K(hx−xi)

- **Kernel (K)** → usually Gaussian
- **Bandwidth (h)** → controls smoothness

 Pros: Flexible, smooth
 Cons: Slow for large data

---

3. Semi-Parametric Methods

Mix of both ideas.

Gaussian Mixture Model (GMM)

Represents data as a mixture of Gaussians.

p(x)=∑k=1KπkN(x|μk,Σk)p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)p(x)=k=1∑KπkN(x|μk,Σk)

Uses **Expectation-Maximization (EM)** algorithm.

Density Estimation in ML Algorithms

Naive Bayes

- Estimates class-conditional densities
- Uses Gaussian or multinomial PDFs

Anomaly Detection

Low probability = anomaly
Used in:

- fraud detection
- network security

Clustering (GMM)

Clusters based on probability density.

Generative Models

- Sample new data from estimated density

- Used in image/text generation

Bayesian Models

Posterior distributions depend on density estimation.

Simple KDE Example (Python)

```
from sklearn.neighbors import KernelDensity
import numpy as np

X = np.array([[1],[2],[2.5],[3],[4]])
kde = KernelDensity(kernel='gaussian', bandwidth=0.5)
kde.fit(X)

log_density = kde.score_samples([[2]])
print(np.exp(log_density))
```

What is Optimization?

In **data science & machine learning**, optimization means:

Finding the best model parameters that minimize error (loss) or maximize performance.

Mathematically:

min⁡θ L(θ)\min_\theta \; L(\theta)θminL(θ)

where

- θ\thetaθ = model parameters
- LLL = loss / cost function

Why optimization is crucial

Without optimization:

- models won’t learn
- predictions stay poor
- training never converges

Optimization decides:

- how fast learning happens
- how good the final model is
- whether training is stable or not

Core Components of Optimization

1. Objective (Loss) Function

Measures **how wrong** the model is.

Common loss functions:

Problem	Loss Function
Linear Regression	Mean Squared Error (MSE)
Classification	Cross-Entropy
Logistic Regression	Log Loss
SVM	Hinge Loss

---

2. Parameters

- Weights
- Biases

Optimization adjusts these to reduce loss.

---

3. Constraints (optional)

Limits on parameters (used in regularization).

Basic Optimization Algorithms

1. Gradient Descent (MOST IMPORTANT)

Moves parameters in the **opposite direction of the gradient**.

θ=θ−α∇L(θ)\theta = \theta - \alpha \nabla L(\theta)θ=θ−α∇L(θ)

- α\alphaα = learning rate

 Types:

- **Batch GD** – whole dataset
- **Stochastic GD (SGD)** – one sample
- **Mini-batch GD** – small batches (most used)

---

2. Stochastic Gradient Descent (SGD)

- Faster updates
- Noisy but escapes local minima

- Used in deep learning

---

3. Momentum

Adds “velocity” to gradient descent.

 Helps:

- reduce oscillations
- speed up convergence

1. Online Learning

What is Online Learning?

Online learning is a training approach where the model **learns continuously**, one data point (or small batch) at a time, instead of training once on the full dataset.

Think: **streaming data, real-time updates**

Why Online Learning is needed

- Data arrives continuously (logs, clicks, IoT)
- Dataset is too large to store
- Data distribution changes over time (**concept drift**)

How Online Learning works

1. Receive a new data point
2. Make a prediction
3. Calculate loss
4. Update model immediately

θt+1=θt−α∇L(θt)\theta_{t+1} = \theta_t - \alpha \nabla L(\theta_t)θt+1=θt−α∇L(θt)

Algorithms that support Online Learning

- Stochastic Gradient Descent (SGD)
- Perceptron
- Online Logistic Regression
- Passive–Aggressive algorithms
- Hoeffding Trees

Online Learning Examples

- Stock price prediction
- Fraud detection
- Recommendation systems
- Real-time ad click prediction

2. Boosting

What is Boosting?

Boosting is an **ensemble learning technique** that combines many **weak learners** to create a **strong learner**.

Idea: *Fix mistakes made by previous models* □

How Boosting works

1. Train a weak model
2. Increase weight of misclassified points
3. Train next model focusing on errors
4. Combine all models (weighted vote)

Popular Boosting Algorithms

□ AdaBoost

- Adjusts data weights
- Focuses on misclassified samples

□ Gradient Boosting

- Fits new model on **residual errors**
- Uses gradient descent idea

□ XGBoost / LightGBM / CatBoost

- Optimized, scalable versions
- Widely used in industry & Kaggle □

Boosting Examples

- Credit scoring

- Fraud detection
- Ranking problems
-

What are Conditional Densities?

A **conditional density** describes the probability distribution of a continuous random variable **given** that another variable has a certain value.

In simple words:

"How likely is *X* **when we already know Y**?"

Mathematical Definition

For continuous variables XXX and YYY:

p(x|y)=p(x,y)p(y)p(x \mid y) = \frac{p(x, y)}{p(y)}p(x|y)=p(y)p(x,y)

Where:

- p(x,y)p(x, y)p(x,y) = joint density
- p(y)p(y)p(y) = marginal density of YYY

This is the continuous version of conditional probability.

Why Conditional Densities Matter in DS & ML

They allow us to:

- make predictions given features
- model uncertainty
- perform probabilistic inference
- build generative and Bayesian models

Almost all supervised ML is about learning:

p(y|x)p(y \mid x)p(y|x)

Conditional Density vs Conditional Probability

- **Discrete data** → conditional probability P(Y|X)P(Y \mid X)P(Y|X)
- **Continuous data** → conditional density p(y|x)p(y \mid x)p(y|x)

Same idea, different math.

Conditional Densities in Machine Learning

1. Supervised Learning (Core Idea)

Given input XXX, predict output YYY:

y^=arg⁡max⁡yp(y|x)\hat{y} = \arg\max_y p(y \mid x)y^=argymaxp(y|x)

Examples:

- Classification → class posterior density
- Regression → conditional distribution of target

---

2. Naive Bayes Classifier

Uses:

p(y|x)∝p(x|y) p(y)p(y \mid x) \propto p(x \mid y)\, p(y)p(y|x)∝p(x|y)p(y)

Here:

- p(x|y)p(x \mid y)p(x|y) = **conditional density of features given class**

---

3. Gaussian Conditional Density

Assume:

X|Y=y∼N(μy,σy2)X \mid Y=y \sim \mathcal{N}(\mu_y, \sigma_y^2)X|Y=y∼N(μy,σy2)

Used in:

- Gaussian Naive Bayes
- Linear Discriminant Analysis (LDA)

---

4. Linear Regression (Probabilistic View)

p(y|x)=N(wTx,σ2)p(y \mid x) = \mathcal{N}(w^Tx, \sigma^2)p(y|x)=N(wTx,σ2)

Prediction = **mean of conditional density**

---

5. Gaussian Mixture Models (GMM)

Each component models:

p(x|z=k)p(x \mid z=k)p(x|z=k)

Used for:

- clustering
- density-based classification

---

6. Bayesian Models

Posterior distribution:

p(θ|x)=p(x|θ)p(θ)p(x)p(\theta \mid x) = \frac{p(x \mid \theta)p(\theta)}{p(x)}p(θ|x)=p(x)p(x|θ)p(θ)

Conditional densities are the backbone of Bayesian inference.

Estimating Conditional Densities

Parametric

- Assume Gaussian, Poisson, etc.
- Estimate parameters from data

Non-Parametric

- Kernel Density Estimation (KDE)
- kNN-based density estimation

Conditional Density Estimation (CDE)

Models p(y|x)p(y \mid x)p(y|x) directly:

- Mixture Density Networks
- Normalizing Flows
- Quantile Regression Forests

Simple Example (Intuition)

Dataset:

- XXX = hours studied
- YYY = exam score

Conditional density:

“What does the **distribution of scores** look like **given 5 hours of study**?”

Not just a point prediction, but a **range with probabilities**.

1. Function Spaces

What is a Function Space?

A **function space** is a set of functions that share certain properties.

Think:

Instead of choosing a single function, ML chooses the **best function from a space of functions**.

Examples of Function Spaces in ML

✅ Linear Function Space

f(x)=wTxf(x) = w^Txf(x)=wTx

Used in:

- Linear regression
- Logistic regression

❌ Limited expressiveness (only straight lines / planes).

✅ Polynomial Function Space

f(x)=w0+w1x+w2x2+…f(x) = w_0 + w_1x + w_2x^2 + \dotsf(x)=w0+w1x+w2x2+…

Used in:

- Polynomial regression

❌ More flexible, risk of overfitting.

✅ Hilbert Space (Advanced but important)

A function space with:

- inner product
- distance
- notion of orthogonality

❌ Many ML models live in **Hilbert spaces**.

Why Function Spaces Matter

They control:

- model capacity
- bias–variance tradeoff
- generalization ability

Choosing a function space = choosing **what the model is allowed to learn**.

---

2. Kernels

What is a Kernel?

A **kernel** is a function that measures **similarity** between two data points:

K(x,x′)=⟨ϕ(x),ϕ(x′)⟩K(x, x') = \langle \phi(x), \phi(x') \rangleK(x,x′)=⟨ϕ(x),ϕ(x′)⟩

It computes an inner product in a **high-dimensional feature space** *without explicitly mapping the data*.

This is the **kernel trick** □□.

Why Kernels are powerful

- Make **non-linear problems linear**
- Avoid explicit feature expansion
- Work efficiently in high dimensions

What is a Linear Model?

A **linear model** assumes the output is a **linear combination of input features**.

y=w0+w1x1+w2x2+⋯+wnxny = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_ny=w0+w1x1+w2x2+⋯+wnxn

Where:

- xix_ixi = features
- wiw_iwi = weights (learned from data)
- w0w_0w0 = bias

Why Linear Models are Important

- Simple and interpretable
- Fast to train
- Strong baseline for any ML problem
- Foundation of many advanced models

Types of Linear Models

1. Linear Regression

 Used for: Continuous output

Goal:
Minimize Mean Squared Error (MSE)

L=1n∑(y−y^)2L = \frac{1}{n}\sum (y - \hat{y})^2L=n1∑(y−y^)2

Examples:

- House price prediction
- Sales forecasting

---

2. Logistic Regression (Linear Classifier)

 Used for: Classification

Model:

p(y=1|x)=σ(wTx)p(y=1|x) = \sigma(w^Tx)p(y=1|x)=σ(wTx)

Uses **sigmoid function** to map output to probability.

Examples:

- Spam detection
- Disease prediction

---

3. Ridge Regression (L2 Regularization)

Adds penalty to large weights:

L=MSE+λ∑w2L = \text{MSE} + \lambda \sum w^2L=MSE+λ∑w2

 Reduces overfitting.

---

4. Lasso Regression (L1 Regularization)
L=MSE+λ∑|w|L = \text{MSE} + \lambda \sum |w|L=MSE+λ∑|w|

 Performs **feature selection**.

---

5. Elastic Net

Combination of L1 + L2.

Linear Models in Classification

Perceptron

- Oldest linear classifier
- Uses sign of wTxw^TxwTx

Linear SVM

- Maximizes margin
- Robust to noise

Geometric Interpretation

- In 2D → line
- In 3D → plane
- In higher dimensions → hyperplane

Linear models **separate data using a straight boundary**.

Optimization in Linear Models

Weights are learned using:

- Normal Equation
- Gradient Descent
- Stochastic Gradient Descent

Probabilistic View

Linear regression assumes:

y|x∼N(wTx,σ2)y|x \sim \mathcal{N}(w^Tx, \sigma^2)y|x∼N(wTx,σ2)

Logistic regression assumes:

y|x∼Bernoulli(σ(wTx))y|x \sim \text{Bernoulli}(\sigma(w^Tx))y|x∼Bernoulli(σ(wTx))

Advantages & Limitations

✅ Advantages

- Interpretable
- Works well on small data
- Low computational cost

❌ Limitations

- Cannot model complex non-linear relationships
- Sensitive to outliers
- Requires feature engineering

Thank you !