**Here's a beginner-friendly overview and code examples for three core Machine Learning algorithms:**

---

## 🤖 Machine Learning Algorithms

---

### 🔹 1. K-Nearest Neighbors (KNN) — *Supervised Learning*

✅ **What it does:**

- **Classifies new data points based on the majority class of its k nearest neighbors in the training data.**
- **Works well for classification and regression (commonly used for classification).**

✅ **How it works:**

- **Calculate distance (e.g., Euclidean) from the test point to all training points.**
- **Pick the k closest points.**
- **Assign the class that is most common among these k.**

✅ **Example (using sklearn):**

```
from sklearn.neighbors import KNeighborsClassifier

# Sample data
X = [[1, 2], [2, 3], [3, 3], [6, 5], [7, 7]]
y = ['A', 'A', 'A', 'B', 'B']

# Train the model
model = KNeighborsClassifier(n_neighbors=3)
model.fit(X, y)

# Predict new value
print("Predicted Class:", model.predict([[4, 4]]))  # Should classify between A and B
```

---

### 🔹 2. K-Means Clustering — *Unsupervised Learning*

✅ **What it does:**

- **Groups data into k clusters where each point belongs to the cluster with the nearest mean (centroid).**

- **Unlabeled data.**

✅ **How it works:**

- **Randomly select k centroids.**
- **Assign points to the nearest centroid.**
- **Recompute centroids.**
- **Repeat until convergence.**

✅ **Example:**

```
from sklearn.cluster import KMeans
import numpy as np

# Sample data
X = np.array([[1, 2], [1, 4], [1, 0],
              [10, 2], [10, 4], [10, 0]])

# Apply KMeans
kmeans = KMeans(n_clusters=2, random_state=0)
kmeans.fit(X)

print("Cluster Centers:\n", kmeans.cluster_centers_)
print("Labels:", kmeans.labels_)
```

---

🔹 **3. Linear Regression — *Supervised Learning***

✅ **What it does:**

- **Models the relationship between independent variable(s) X and a dependent variable y using a straight line.**
- **Predicts continuous values.**

✅ **How it works:**

- **Finds the best-fit line:**

**$y=mx+c$ y = mx + c**

✅ **Example:**

```
from sklearn.linear_model import LinearRegression
```

```
import numpy as np

# Training data
X = np.array([[1], [2], [3], [4], [5]])
y = np.array([2, 4, 5, 4, 5])

# Train the model
model = LinearRegression()
model.fit(X, y)

# Predict
print("Predicted:", model.predict([[6]]))  # Predict y when x=6
```

---

✅ **Summary Table**

| Algorithm | Type | Use Case | Output Type |
|---|---|---|---|
| KNN | Supervised | Classification | Class Label |
| K-Means | Unsupervised | Grouping / Segmentation | Cluster ID |
| Linear Regression | Supervised | Predicting Values | Continuous |

---

**Would you like a small project that combines these algorithms using real-world data (e.g., Titanic, Iris, or customer segments)?**