🚀 Understanding Document Vectors with BERT & K-Means Clustering

(Explained in the simplest way using email classification as an example!)

Imagine you receive **hundreds of emails** every week—some from your university, some about exams, and some from your favorite online store with discounts. Wouldn't it be great if your computer could **automatically sort these emails into different categories**?

That's exactly what we can do with **BERT embeddings** and **K-Means Clustering**! Let's go step by step.

🟢 Step 1: Understanding the Problem

💡 You receive emails every day. Some are about:
1️⃣ **University** (Assignments, deadlines, schedules)
2️⃣ **Shopping** (Amazon, Flipkart discounts)
3️⃣ **Banking** (Statements, transactions)
4️⃣ **Spam** (Unknown lottery wins, fake offers)

- 🔹 Right now, you manually **read each email** to decide where it belongs.
- 🔹 But what if a machine could **understand** the email and **automatically sort it**?

This is what we will do using **BERT embeddings** (to understand emails) and **K-Means Clustering** (to group similar emails).

🟢 Step 2: What is a Document Vector?

💡 Computers don't understand words, only numbers.
📧 So, before we can process emails, we must **convert them into numbers** (vectors).

Let's take an example email:

> **"Your assignment is due tomorrow. Submit it before 5 PM."**

- 🔹 The computer does not understand this sentence.
- 🔹 We use **BERT** (a special AI model) to **convert this email into a list of numbers** (a document vector).

🔹 **Example Output from BERT** (simplified):

[0.12, -0.87, 0.54, 0.75, ..., 0.23] (Total 768 numbers)

Each email will now be represented as a list of **768 numbers** instead of words.

🟢 Step 3: Using BERT to Get Document Vectors

BERT (Bidirectional Encoder Representations from Transformers) is a special AI model that reads text like a human.

📌 **How BERT works:**

- It understands words **in context** (so "bank" in *money bank* is different from *river bank*).
- It converts an email into a **768-dimensional vector** (a long list of numbers).

🖍️ **Example:** Let's say we have 3 emails:

1️⃣ **"Your assignment is due tomorrow. Submit it before 5 PM."**
2️⃣ **"Your Amazon order will be delivered tomorrow."**
3️⃣ **"Your bank statement for January is ready."**

BERT converts them into **document vectors** like this:

Email 1 (University) → [0.12, -0.87, 0.54, ..., 0.23]
Email 2 (Shopping) → [-0.55, 0.87, -0.23, ..., -0.10]
Email 3 (Banking) → [0.67, -0.43, 0.88, ..., -0.30]

Now, emails are **just numbers** (vectors), and similar emails will have **similar numbers**.

🟢 Step 4: How K-Means Clustering Works

Now that all emails are **converted into numbers**, we can **group similar emails together** using **K-Means Clustering**.

💡 **Think of K-Means like sorting school kids into study groups based on their favorite subjects.**
📍 It finds similar emails and puts them into the same group!

🟢 Step 5: Steps in K-Means Clustering

🔵 Step 1: Choose the Number of Clusters (K)

Before starting, we decide how many groups (clusters) we want.
For example:

- **K = 3** → We want 3 groups (University, Shopping, Banking).
- **K = 4** → We want 4 groups (University, Shopping, Banking, Spam).

🔵 Step 2: Pick K Random Emails as Starting Points (Centroids)

🔹 K-Means randomly picks **K emails** as the "starting points" for each group.
🔹 These are called **centroids** (the center of a cluster).

📌 **Example:**
If we have **K = 3** clusters:

- Centroid 1 → Randomly picks a **University email**.
- Centroid 2 → Randomly picks a **Shopping email**.
- Centroid 3 → Randomly picks a **Banking email**.

🔵 Step 3: Assign Each Email to the Nearest Centroid

Now, for each new email:
1️⃣ The computer checks **which centroid is closest**.
2️⃣ It assigns the email to that group.

🔵 Step 4: Update Centroids

After grouping all emails, we **recalculate** the centroid by averaging all the document vectors in each group.

🔵 Step 5: Repeat Steps 3 & 4 Until Clusters Stop Changing

The algorithm **keeps repeating** Steps 3 and 4:
1️⃣ Assigning emails to the nearest centroid.
2️⃣ Updating the centroid position.
3️⃣ Repeating this until **the centroids stop moving** (meaning the groups are stable).

🟢 Step 6: Result - Grouping the Emails

🔹 After running the K-Means algorithm, we get **well-separated email clusters** like this:

📂 **Cluster 1: University Emails**
📂 **Cluster 2: Shopping Emails**
📂 **Cluster 3: Banking Emails**
📂 **Cluster 4: Spam Emails**

Now, all similar emails are **grouped together automatically**! 🎉

📌 Final Summary

💯 **BERT** converts emails into **document vectors (numbers).**
💯 **K-Means** finds patterns and groups similar emails together.
💯 **Clusters form naturally** based on topic similarity.

💡 **Now, your inbox is automatically organized!**