Name:Richa Patel
Course: COSC 522
Module: M5

Question 1:

What are the sizes of your training, validation, and test set?

Answer 1:

After splitting the dataset, the sizes of each set are:

- **Training set:** 951 samples
- **Validation set:** 204 samples
- **Test set:** 204 samples

This split follows a 70% training, 15% validation, and 15% test strategy, which helps ensure that the model has enough data to learn from, tune on, and evaluate generalization performance.

Question 2:

Which features did you choose? For each feature, do you have intuition at why it might impact the final price of phones?

Answer 2:

I selected 10 features for my model that I believe are closely related to how phone prices vary. Here's what I picked and why:

1. **Battery capacity (mAh)** – Phones with higher battery capacity usually offer longer usage, which is a selling point for many users, so they tend to cost more.
2. **Screen size (inches)** – Larger screens usually mean better display experience, which is typically found in higher-end phones.
3. **Resolution x and Resolution y** – Higher resolution means sharper and better-quality displays, which usually adds to the phone's cost.

4. **RAM (MB)** – More RAM means better multitasking and faster performance. High-RAM phones are generally more expensive.
5. **Internal storage (GB)** – Phones with more internal storage let users store more apps, videos, and files, so these models are usually priced higher.
6. **Rear camera** – A higher megapixel rear camera generally means better photo quality, which often comes with a higher price tag.
7. **Front camera** – Especially with the popularity of selfies and video calls, a better front camera adds to the value and cost of the phone.
8. **Touchscreen** – Most modern phones have a touchscreen, but it's still an important feature that differentiates basic phones from smartphones.
9. **4G/ LTE** – Phones that support 4G or LTE are newer and offer faster internet speeds, which usually means they cost more than 3G phones.

Overall, I tried to choose features that affect performance, usability, and user experience, since those are the things that usually make phones more expensive.

Question 3:

How did you normalize your data?

Answer 3:

I normalized the data using **MinMaxScaler** from the `sklearn.preprocessing` module. I applied it to the numeric features only — like battery capacity, screen size, resolution, RAM, storage, and both camera specs.

The scaler transforms all the values to be between 0 and 1. I didn't apply it to the one-hot encoded columns (like `Touchscreen_Yes` and `4G/ LTE_Yes`) because they are already binary (0 or 1), and scaling them would not make sense.

Normalization was important because the features had very different ranges. For example, RAM was in thousands, while screen size was just a single-digit number. Scaling them helps the neural network train more effectively, without letting any single large-valued feature dominate the learning process.

Question 4:

Describe your early and show its training. How do you know that it is working / over training?

Answer 4:

My early model had two hidden layers with 128 neurons each and used ReLU as the activation function. I trained it for 100 epochs without using early stopping, to observe how the model performs when overtrained. During training, the training MAE kept decreasing, reaching around ₹5,900, which seemed good. However, the validation MAE started increasing, going above ₹6,800, which indicated that the model was starting to memorize the training data instead of learning general patterns. This increasing gap between training and validation performance was a clear sign of overfitting. That's how I knew the model was overtraining and not generalizing well.

Question 5:

At what number of epochs did you decide to stop your "final" model?

Answer 5:

I used early stopping in my final model to monitor validation loss.
The training continued for all 100 epochs because validation loss kept improving.
Early stopping didn't stop the training early, but it still kept the best model weights.
This helped prevent overfitting and made sure the final model generalized well.
It ensured I didn't end up using a model from a worse-performing epoch.

Question 6:

Describe your "final" model and show its training performance.

Answer 6:

My final model had two hidden layers with 64 neurons each and used ReLU activation. I used the Adam optimizer and mean squared error as the loss function. I also added early stopping to prevent overfitting. The training ran for 100 epochs, but early stopping kept the best model weights.

The final training MAE was around ₹6,401 and the validation MAE was about ₹7,425. These values were closer together compared to the early overfitted model, showing better generalization and more stable training performance.

Question 7:

What is the final performance metric you chose and how do you feel the model performed? Did it match your expectations?

Answer 7:

I chose **Mean Absolute Error (MAE)** as my final performance metric because it's easy to interpret — it tells me how far off my predictions are, on average. My final model achieved a test MAE of about **₹6,078**, which means the predicted prices were usually within ₹6,000 of the actual values. For a basic deep learning model using only 10 features, I think this is a good result. It matched my expectations for a first attempt and showed that the model generalized well without overfitting.

Question 8:

With additional time, is there anything you would have done differently?

Answer 8:

Yes, with more time, I would have explored different feature combinations and tried adding or removing some to see how they affect the model. I’d also experiment with changing the number of layers or neurons to improve performance. Trying different activation functions like `tanh` and adjusting early stopping settings could have helped. Finally, I would spend more time analyzing model errors to understand where it's making the biggest mistakes.

BONUS:

Are there any summative insights you have drawn from working with this data?

Yes, working with this dataset helped me understand how different phone specifications can influence the price. Features like RAM, storage, and battery capacity had a clear impact, which made sense based on real-world experience. I also learned how important it is to prepare data properly — like encoding and normalization — before training a model. Seeing how overfitting affects performance and how early stopping helps avoid it was also a big takeaway. Overall, this project gave me a clear understanding of how deep learning can be applied to real-world regression problems.