----------REMINDER----------

If training MAPE is 30% for example and test MAPE is 30.5% then their difference is 0.5%.

So since 0.5% is < 10% (benchmark for overfitting) we conclude that the model does not overfit.

A reason for overfitting is that you are trying to fit a too complex model to a small dataset.

Since test MAPE is 30%, do we accept the model for forecasts? The answer is it

depends on the benchmark we have for test MAPE.

Since test MAPE is e.g., 30.5% then we say that the model accuracy is expected to have an error of around 30%

on the forecasts. We will never learn the forecast error since we will never learn the future (until it happens).

So, at the moment we can have a proxy of forecast error only.

So, we check 2 things:

a) First the test MAPE on its own.

b) the difference test MAPE minus train MAPE must be <= 10% or there is overfitting.

So, if both are ok, then we can use the model for forecasts. But if either is bad, then we reject the model.

==========

If we had more data, then we would be doing hyperparameter tuning:

a) split data into training, validation and test sets: training set: 60% of the data, validation set: 20% of the data, test set: 20% of the data.

b) fit the model to the training set with different polynomial degrees (e.g., 2, 3, 4), and calculate the error on the validation set;

c) find the hyperparameter combination that minimizes the validation error. We say that we "tune" the hyperparameters on the validation set. Eg select the best polynomial degree as the one with the smallest validation error;

d) fit the model to the combined training and validation sets using this best hyperparameter (polynomial degree), and calculate the error on the test set ie the test MAPE for that hyperparameter combination.

Tuning the hyperparameters is not mandatory. The fact that we didn't tune the hyperparameters doesn't make the model outputs meaningless.

In our case we selected the hyperparameters based on our own intuition, typical values or previous research.

In the sensitivity analysis, we find the test MAPE based on different combos of hyperparameters.

We are not tuning the hyperparameters meaning that we do not claim that we find optimal hyperparameters.

We just simply observe how the test MAPE changes with some randomly chosen hyperparameter values around the ones we chose

in the basic study. Just to get an idea of the model behaviour around the neighbourhood of values we selected in the beginning.

Tuning hyperparameters does not involve calculating the test MAPE as described above. It involves finding the error on a validation set. Here, we do not have a validation set.

The test set should only be used for calculating the test MAPE. And not to find optimal hyperparameters.
Because if you use the test set for finding the optimal hyperparameters, then this means that the next step is to train the model using these optimal hyperparameters. And then finding the test MAPE.
In this case the test MAPE would not be a proxy of the forecasting error though because we would have used the test set to find optimal hyperparameters. IE in that case a good test set performance (i.e., small test MAPE) doesn't necessarily translate into good future performance.

The test MAPE is a good proxy of the forecasting error if and only if the test set is not used in any way for building the model.

The idea for the test set is to be an unseen set.
You want to approximate how the model will perform in the future, on new unseen data.

So, what difference would it make if the below forecasts were done using tuned hyperparameters?
Answer: If the tuning is done properly, i.e., with a training-validation-test split and not with a training-test split, then the model outputs are somewhat more reliable as they are generated by a data-driven model (i.e., using the best model configuration for the given dataset) rather than an arbitrary set of hyperparameters.