Reference: https://www.kaggle.com/alexisbcook/data-leakage

- **Data leakage or leakage**
    - **What/When it happen ?**

Happens when your training data contains information about the target, but similar data will not be available when the model is used for prediction. This leads to high performance on the training set (and possibly even the validation data), but the model will perform poorly in production.

- **Why is it important ?**

Leakage causes a model to look accurate until you start making decisions with the model, and then the model becomes very inaccurate.

- **Type of data leakage ?**

Two main types:

- Target leakage
- Train-test contamination

1. **Target leakage**

Occurs when your predictors include data that will not be available at the time you make predictions. It is important to think about target leakage in terms of the *timing or chronological order* that data becomes available, not merely whether a feature helps make good predictions.

- **To prevent this type of data leakage, any variable updated (or created) after the target value is realized should be excluded.**

2. Train-Test Contamination

A different type of leak occurs when you aren't careful to distinguish training data from validation data. Recall that validation is meant to be a measure of how the model does on data that it hasn't considered before. You can corrupt this process in subtle ways if the validation data affects the preprocessing behavior. This is sometimes called train-test contamination.

- If your validation is based on a simple train-test split, exclude the validation data from any type of *fitting*, including the fitting of preprocessing steps. This is easier if you use scikit-learn pipelines. When using cross-validation, it's even more critical that you do your preprocessing inside the pipeline!

Resources:

- https://machinelearningmastery.com/data-leakage-machine-learning/