Section 1: What Is Feature Engineering?

- The goal of feature engineering is simply to make your data better suited to the problem at hand.
- We might perform feature engineering to:
    - Improve a models predictive performance
    - Reduce computational or data needs
    - Improve interpretability of the result
- For a feature to be useful, it must have a relationship to the target that your model is able to learn.
- Linear models, for instance, are only able to learn linear relationships. So, when using a linear model, our goal is to transform the features to make their relationship to the target linear.
- Engineering new features can improve model performance.

Section 2 : Mutual Information

- A great first step in feature engineering is to construct a ranking with a feature utility metric, a function measuring associations between a feature and the target. Then we can choose a smaller set of the most useful features to develop initially.
- Mutual information is the metric that we will use here.
- Mutual information is a lot like correlation in that it measures a relationship between two quantities. The advantage of mutual information is that it can detect any kind of relationship, while correlation only detects linear relationships.
- Mutual information is a great general-purpose metric and especially useful at the start of feature development when we might not know what model we would like to use.

- Mutual information metrics is:
    - Easy to use and interpret,
    - Computationally efficient,
    - Theoretically well-founded,
    - Resistant to overfitting, and,
    - Able to detect any kind of relationship
- Mutual information describes relationships in terms of uncertainty.
- The mutual information (MI) between two quantities is a measure of the extent to which knowledge one quantity reduces uncertainty about the other.
- The least possible mutual information between quantities is 0. When MI is zero, the quantities are independent: neither can tell you anything about the other. Conversely, in theory there’s no upper bound to what MI can be. In practice though values above 2 or so are uncommon. (Mutual information is a logarithmic quantity, so it increases very slowly.)
- Here are some things to remember when applying mutual information:
    - MI can help you to understand the relative potential of a feature as a predictor of the target, considered by itself.
    - It’s possible for a feature to be very informative when interacting with other features, but not so informative all alone.
    - MI can’t detect interactions between features. It is a univariate metric.
    - Just because a feature has a high MI score doesn’t mean your model will be able to do anything with that information. You may need to transform the feature first to expose the association.

- The scikit-learn algorithm for MI treats discrete features differently from continuous features. Consequently, you need to tell it which are which. As a rule of thumb, anything that must have a float dtype is not discrete. Categoricals(object or categorical dtype) can be treated as discrete by giving them a label encoding.
- Scikit-learn has two mutual information metrics in its feature_selection module : one for real-valued targets(mutual_info_regression) and one for categorical targets(mutual_info_classif).
- Before deciding a feature is unimportant from its MI score, it’s good to investigate any possible interaction effects – domain knowledge can offer a lot of guidance here.
- Data visualization is a great addition to your feature-engineering toolbox. Along with utility metrics like mutual information, visualizations like these can help you discover important relationships in your data.