**Pipelines**

Estimator / algorithm – using fit() –> Transformer / Model

Example:

* From estimator to learned models
  + LogisticRegression -> LogisticRegressionModel
  + Pipeline -> PipelineModel
    - Assuming the pipeline includes at least one estimator
* Feature Transformers
* Tokenizer (is a transformer)
* HashingTF (is a transformer)

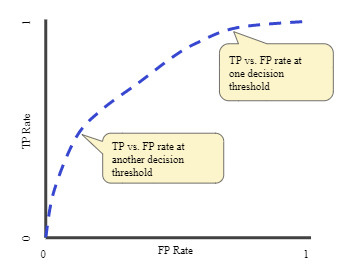
Data Frame – Transformer/Model.transform() –> Data Frame

**Model selection and tuning**

Three required components for *CrossValidator()* and *TrainValidationSplit()*

1. Estimator: It can be a pipeline
2. ParamGridBuilder: *ParamGridBuilder()*
3. Evaluator: *RegressionEvaluator(), BinaryClassifictionEvaluator(), MulticlassClassficiationEvaluator()*

After fitting CrossValidator and TrainValidationSplit object, we will get a **learned model** with parameters that generate the best result.

**Evaluation Metrics**

ROC (Receiver Operating Characteristic) plots TP Rate against FP Rate.

* TP Rate = TP / T = TP / (TP + FN)
  + Note: When FN is small, TP Rate is large. We want TPR to be large all the time.
* FP Rate = FP / F = FP / (FP + TN)
  + Note: When TN is large, FP Rate is small. We want FPR to be small all the time.

One very important concept of ROC is ***ranking*** of the prediction. All predictions, including 0 and 1, are ranked by probability from high to low, and then the TPR and FPR are calculated as the probability goes down. This process generates the curve that we see in the chart.

***Threshold tuning*** is a thing. Depending on the use case, we might want to adjust TP/FP by changing the threshold.

Source of chart: <https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc>