Week 2

Additional material and links

Visualization tools

- Seaborn
- Plotly
- Bokeh
- ggplot
- Graph visualization with NetworkX

Others

- Biclustering algorithms for sorting corrplots

Validation strategies

This page contains information about main validation strategies (schemes): **holdout**, **K-Fold**, **LOO**.

The main rule you should know — *never use data you train on to measure the quality of your model*. The trick is to split all your data into *training* and *validation* parts.

Below you will find several ways to validate a model.

a) Holdout scheme:

1. Split train data into two parts: partA and partB.
2. Fit the model on partA, predict for partB.
3. Use predictions for partB for estimating model quality. Find such hyper-parameters, that quality on partB is maximized.

b) K-Fold scheme:

1. Split train data into K folds.
2. Iterate though each fold: retrain the model on all folds except current fold, predict for the current fold.
3. Use the predictions to calculate quality on each fold. Find such hyper-parameters, that quality on each fold is maximized. You can also estimate mean and variance of the loss. This is very helpful in order to understand significance of improvement.

c) LOO (Leave-One-Out) scheme:

1. Iterate over samples: retrain the model on all samples except current sample, predict for the current sample. You will need to retrain the model N times (if N is the number of samples in the dataset).
2. In the end you will get LOO predictions for every sample in the trainset and can calculate loss.

Additional material and links

- Validation in Sklearn
- Advices on validation in a competition

Additional material and links

- Perfect score script by Oleg Trott -- used to probe leaderboard
- Page about data leakages on Kaggle
- Another page about data leakages on Kaggle

Week 3

Additional material and links

Visualization tools

- Seaborn
- Plotly
- Bokeh
- ggplot
- Graph visualization with NetworkX

Others

- Biclustering algorithms for sorting corrplots

Week 3

Classification

- Evaluation Metrics for Classification Problems: Quick Examples + References
- Decision Trees: “Gini” vs. “Entropy” criteria
- Understanding ROC curves

Ranking

- Learning to Rank using Gradient Descent -- original paper about pairwise method for AUC optimization
- Overview of further developments of RankNet
- RankLib (implemtations for the 2 papers from above)
- Learning to Rank Overview

Clustering

- Evaluation metrics for clustering

Week 4

Additional material and links

- Tuning the hyper-parameters of an estimator (sklearn)
- Optimizing hyperparameters with hyperopt
- Complete Guide to Parameter Tuning in Gradient Boosting (GBM) in Python

Additional materials and links

Far0n's framework for Kaggle competitions "kaggletils"

28 Jupyter Notebook tips, tricks and shortcuts

Additional Materials and Links

Matrix Factorization:

- Overview of Matrix Decomposition methods (sklearn)

t-SNE:

- Multicore t-SNE implementation
- Comparison of Manifold Learning methods (sklearn)
- How to Use t-SNE Effectively (distill.pub blog)
- tSNE homepage (Laurens van der Maaten)
- Example: tSNE with different perplexities (sklearn)

Interactions:

- Facebook Research's paper about extracting categorical features from trees
- Example: Feature transformations with ensembles of trees (sklearn)

Week4

Additional materials and links

- Kaggle ensembling guide at MLWave.com (overview of approaches)
- StackNet — a computational, scalable and analytical meta modelling framework (by KazAnova)
- Heamy — a set of useful tools for competitive data science (including ensembling)

Week 5

Additional material and links

You can often find a solution of the competition you're interested on its forum. Here we put links to collections of such solutions that will prove useful to you.

Past solutions

- http://ndres.me/kaggle-past-solutions/
- https://www.kaggle.com/wiki/PastSolutions
- http://www.chioka.in/kaggle-competition-solutions/
- https://github.com/ShuaiW/kaggle-classification/