Hyperparameters tuning
Part I

Plan for the lecture

- Hyperparameter tuning in general
  - General pipeline
  - Manual and automatic tuning
  - What should we understand about hyperparameters?
- Models, libraries and hyperparameter optimization
  - Tree-based models
  - Neural networks
  - Linear models

Plan for the lecture: models

- Tree-based models
  - GBDT: `XGBoost, LightGBM, CatBoost`
  - RandomForest/ExtraTrees
- Neural nets
  - `Pytorch, Tensorflow, Keras...`
- Linear models
  - SVM, logistic regression
  - `Vowpal Wabbit, FTRL`
- Factorization Machines (out of scope)
  - `libFM, libFFM`

How do we tune hyperparameters

How do we tune hyperparameters

1. **Select the most influential parameters**
    a. There are tons of parameters and we can't tune all of them

How do we tune hyperparameters

1. **Select the most influential parameters**
   a. There are tons of parameters and we can't tune all of them
2. **Understand, how exactly they influence the training**

How do we tune hyperparameters

1. **Select the most influential parameters**
   a. There are tons of parameters and we can’t tune all of them
2. **Understand, how exactly they influence the training**
3. **Tune them!**
   a. Manually (change and examine)
   b. Automatically (hyperopt, etc.)

Hyperparameter optimization software

- A lot of libraries to try:
  - *Hyperopt*
  - Scikit-optimize
  - Spearmint
  - GPyOpt
  - RoBO
  - SMAC3

Automatic hyperparameter optimization

```
def xgb_score(param):
    # run XGBoost with parameters `param`

def xgb_hyperopt():
    space = {
          'eta' : 0.01,
          'max_depth' :         hp.quniform('max_depth', 10, 30,1),
          'min_child_weight' : hp.quniform('min_child_weight', 0, 100, 1),
          'subsample' :         hp.quniform('subsample', 0.1, 1.0, 0.1),
          'gamma' :             hp.quniform('gamma', 0.0, 30, 0.5),
          'colsample_bytree' : hp.quniform('colsample_bytree', 0.1, 1.0, 0.1),

          'objective':'reg:linear',

          'nthread' : 28,
          'silent' : 1,
          'num_round' : 2500,
          'seed' : 2441,
          'early_stopping_rounds':100
          }


    best = fmin(xgb_score, space, algo=tpe.suggest, max_evals=1000)
```

Color-coding legend

1. Underfitting (bad)
2. **Good fit and generalization (good)**
3. Overfitting (bad)

Color-coding legend

- A parameter in red
  - *Increasing it impedes fitting*
  - Increase it to reduce overfitting
  - Decrease to allow model fit easier

Color-coding legend

- A parameter in red
  - *Increasing it impedes fitting*
  - Increase it to reduce overfitting
  - Decrease to allow model fit easier
- A parameter in green
  - *Increasing it leads to a better fit (overfit) on train set*
  - Increase it, if model underfits
  - Decrease if overfits

Conclusion

- Hyperparameter tuning in general
  - General pipeline
  - Manual and automatic tuning
  - What should we understand about hyperparameters?

Conclusion

- Hyperparameter tuning in general
  - General pipeline
  - Manual and automatic tuning
  - What should we understand about hyperparameters?
- Models, libraries and hyperparameter optimization
  - Tree-based models
  - Neural networks
  - Linear models