****

**Intro**

This is the first module where you will try to fit the machine learning models by yourself.

To accomplish it you need to know about basic ML pipeline:

1. EDA
2. Choosing quality metrics
3. Feature engineering
4. Train/dev/test split
5. Baseline training - first results
6. Tuning hyperparameters (improving metrics on dev dataset)
7. Watch for under/ and overfitting
8. Adding/modifying features to improve results
9. Back to step 6
10. If satisfied with metric value - good job, keep calm and carry on

These are mandatory steps. Now you're gonna learn some theory about it.

**Regression problem**

<https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/>

Linear regression

<https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86>

Polynomial regression

<https://www.analyticsvidhya.com/blog/2020/03/polynomial-regression-python/>

Residual analysis

<https://stattrek.com/regression/residual-analysis.aspx> (short overview)

<http://docs.statwing.com/interpreting-residual-plots-to-improve-your-regression/#outlier-header> (detailed posts)

**Metrics**

R-squared

[StatQuest: R-squared explained](https://www.youtube.com/watch?v=2AQKmw14mHM&feature=youtu.be)

Adjusted R-squared

<http://thestatsgeek.com/2013/10/28/r-squared-and-adjusted-r-squared/>

Regularization

<https://www.analyticssteps.com/blogs/l2-and-l1-regularization-machine-learning>

**Gradient descent**

For linear regression:

<https://towardsdatascience.com/linear-regression-using-gradient-descent-97a6c8700931>

Method overview:

<https://machinelearningmastery.com/gradient-descent-for-machine-learning/>

GD algorithms

<https://ruder.io/optimizing-gradient-descent/>

required parts:

- Gradient descent variants;

- Challenges;

- Gradient descent optimisation algorithms

other is optional

**Core ML pipeline concepts:**

Model fitting, under- and overfitting, train/test split, cross-validation

<https://jakevdp.github.io/PythonDataScienceHandbook/05.03-hyperparameters-and-model-validation.html>

<https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6>

<https://medium.com/@snji.khjuria/everything-you-need-to-know-about-train-dev-test-split-what-how-and-why-6ca17ea6f35>

Loss-function and quality metric

<https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/>

<https://stackoverflow.com/questions/56634973/why-would-i-choose-a-loss-function-differing-from-my-metrics> (?)

<https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23>

<https://www.analyticsvidhya.com/blog/2019/08/detailed-guide-7-loss-functions-machine-learning-python-code/>

Bias/variance tradeoff

<https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229>

Regularization

<https://www.einfochips.com/blog/regularization-make-your-machine-learning-algorithms-learn-not-memorize/>

**(\*) Optional materials:**

<https://explained.ai/regularization/index.html>- Regularization explained in more details