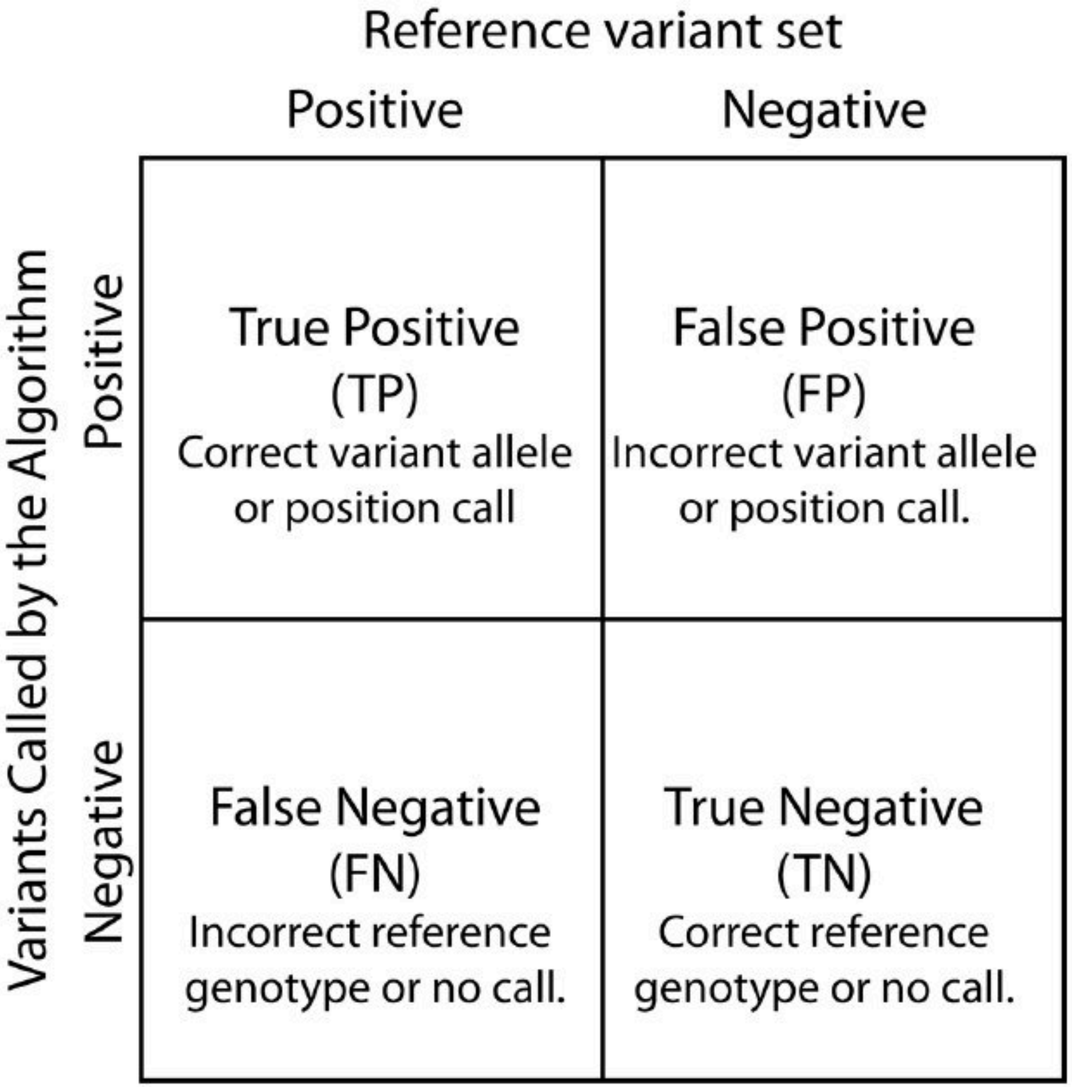
Text classification reminder / Use case

Malyutin Eugeny

Motivation

- **Sentiment analysis**: track/check if the users are happy with the product or not (optional: + find out which particular feature user [dis]liked)
- **Topic classification**: section the news article to be put in
- **Spam detection**: predict if letters are unwanted by user based on those tagged by him/her as spam
- **Incomplete data imputation**: predict user's gender based on text he/she publishes/likes/skips
- **Many more**: authorship attribution, sociodemographic characteristics, etc...

Binary classification quality evaluation



Binary classification quality evaluation

Variants Called by the Algorithm	Reference variant set: Positive	Reference variant set: Negative
Positive	True Positive (TP) Correct variant allele or position call	False Positive (FP) Incorrect variant allele or position call.
Negative	False Negative (FN) Incorrect reference genotype or no call.	True Negative (TN) Correct reference genotype or no call.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

The F-score

The F-score is the Harmonic mean of Precision and Recall.

$$F = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$$

Alternatively

$$F = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

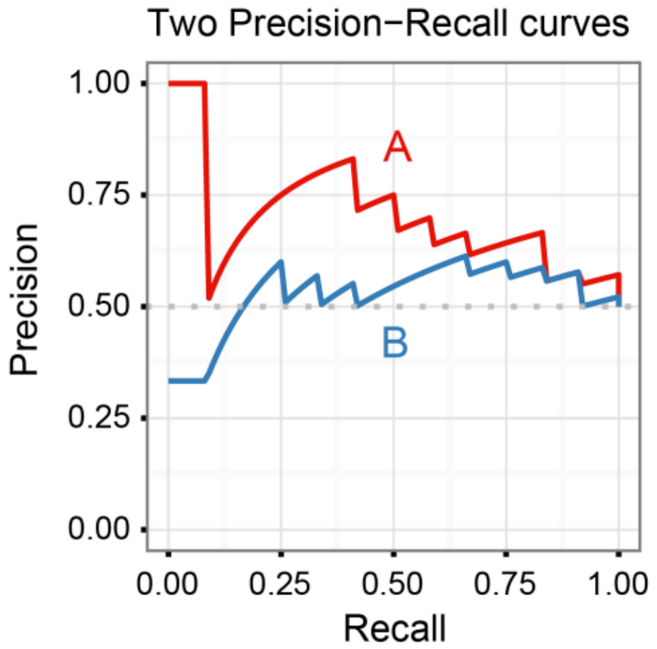
Checking your understanding: if the classifier sets all labels as the target class (all samples are predicted as positive ones), what are precision and recall?

Binary classification quality evaluation

Precision-Recall Curve

we change the parameter that changes precision and recall and look at the behaviour of precision and recall values (this parameter is usually a probability threshold in a decision rule)

For one number without any thresholding - count Area Under Curve - ROC_AUC



Multi-class

= number of classes > 2

- **Accuracy** share of correctly predicted cases

- **Micro-averaging**: Precision, Recall, FScore
  first we compute TP, FP, …, for every class and then we compute metrics values, summing all TPs, FPs, etc.

- **Macro-averaging** aka “all classes are equally important”: Precision, Recall, FScore computing Precision, Recall,...
  for every class, then averaging (summing and dividing by the number of classes)

Texts representation:

- Already discussed: ngrams, count/one-hot/tf-idf/normalized tf-idf
- Custom features may also help: POS counts, text length, weighted average word embeddings, RNN-based embeddings, etc.

the dog is on the table

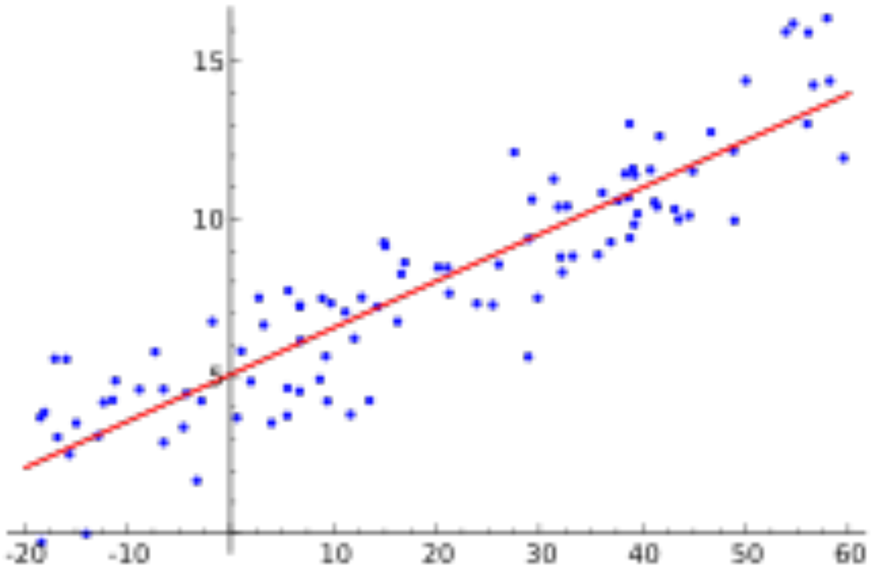
0	0	1	1	0	1	1	1
are	cat	dog	is	now	on	table	the

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

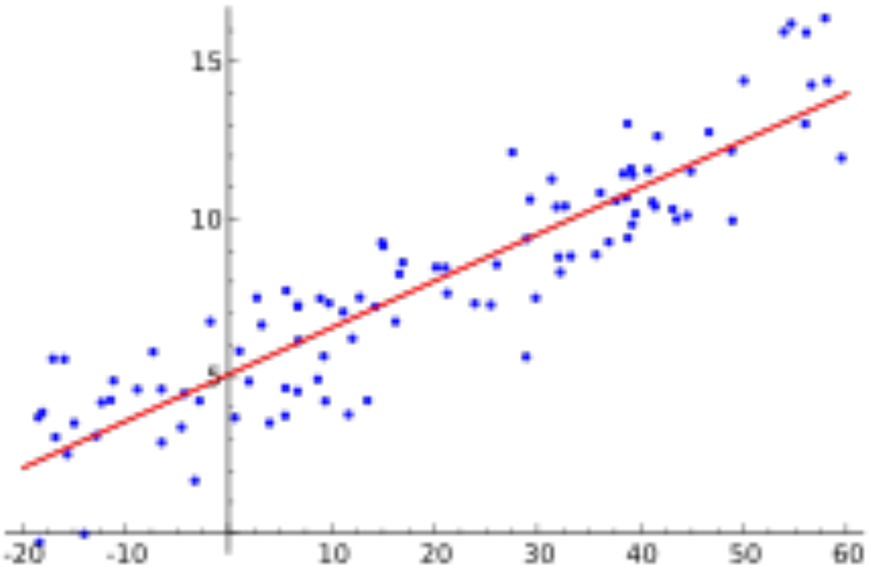
Linear models:

- What is it?
- What if we want to classify?
- What if we want probability?
- What is overfitting and how can we deal with it?
- How should we prepare our data?
- What is the key difference between L1 and L2?



Linear models:

- training and prediction is fast
- more robust then naive Bayes and works better with correlated features
- has to be done: tuning regularization, feature normalization, feature selection
- probabilities estimates may not reflect the data,



SVM

Points of classes **c** from the set { -1, 1 }:

$$\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \ldots, (\mathbf{x}_n, c_n)\}$$

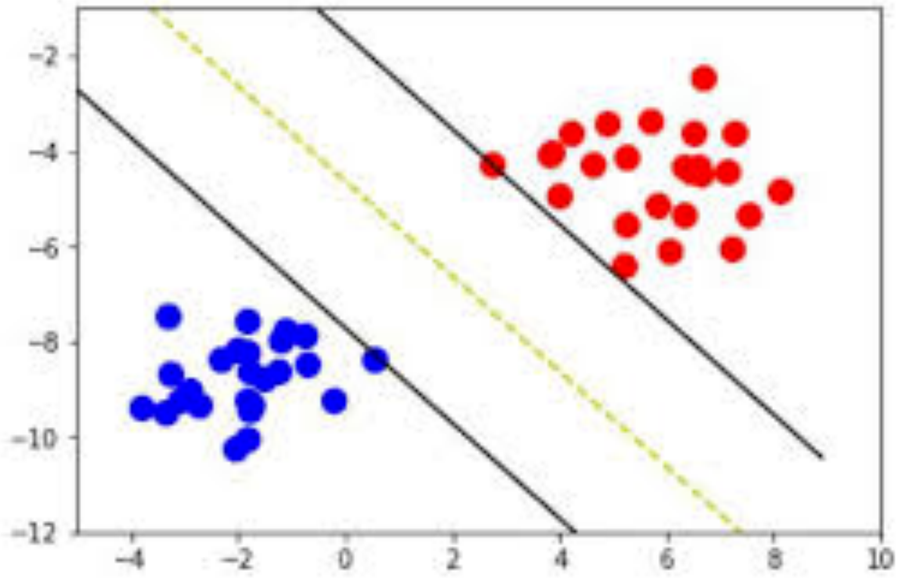
Separating hyperplane:

$$\mathbf{w} \cdot \mathbf{x} - b = 0.$$

two **parallel hyperplanes** that we can move without touching the samples in the case of linear separability:

$$\mathbf{w} \cdot \mathbf{x} - b = 1, \quad \mathbf{w} \cdot \mathbf{x} - b = -1.$$

So we minimize **|w|**, so that the distance between them was greater



SVM

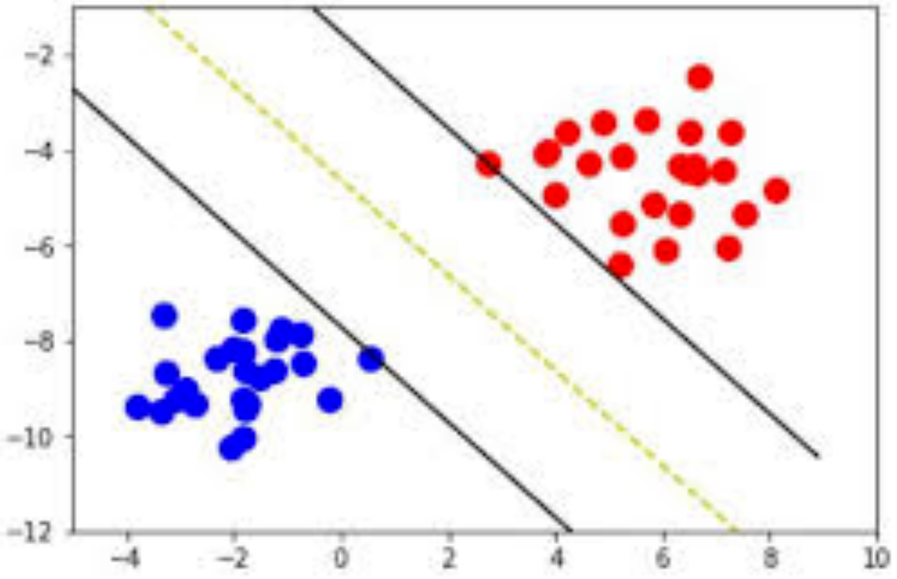
Quadratic programming task

$$\begin{cases} \|\mathbf{w}\|^2 \to \min \\ c_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1, \quad 1 \leq i \leq n. \end{cases}$$

with a few transformations we can reformulate the task like this:

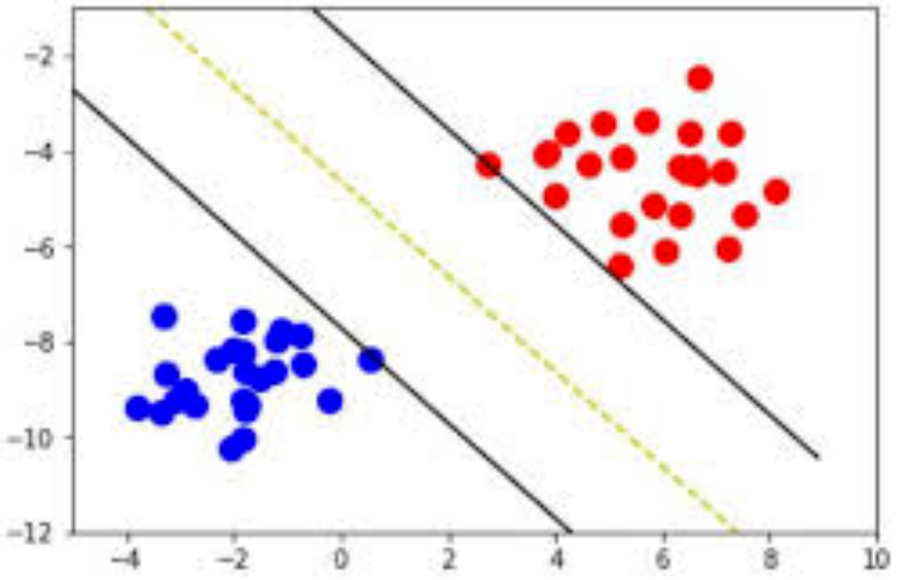
$$\begin{cases} -\mathbf{L}(\lambda) = -\sum_{i=1}^{n} \lambda_\mathbf{i} + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_\mathbf{i} \lambda_\mathbf{j} c_i c_j (\mathbf{x_i} \cdot \mathbf{x_j}) \to \min_\lambda \\ \lambda_\mathbf{i} \geq 0, \quad 1 \leq i \leq n \\ \sum_{i=1}^{n} \lambda_\mathbf{i} c_i = 0 \end{cases}$$

this quadratic programming task has just one solution, which can be effectively found in the case if hundreds of thousands objects



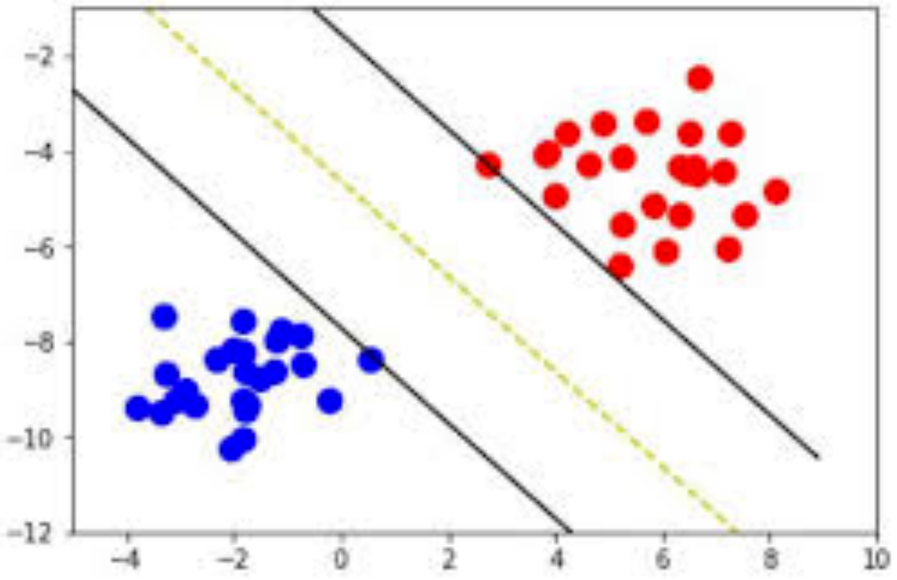
SVM

- Why is it differ from regression? (our algorithm is still linear)
- What is kernel trick?
- Do you know any production SVM use case?



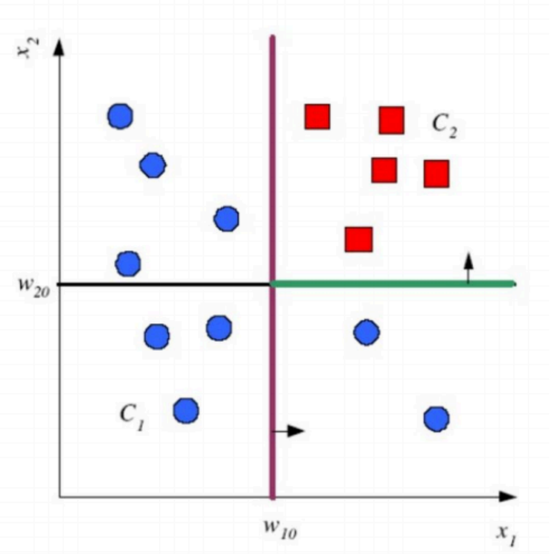
SVM

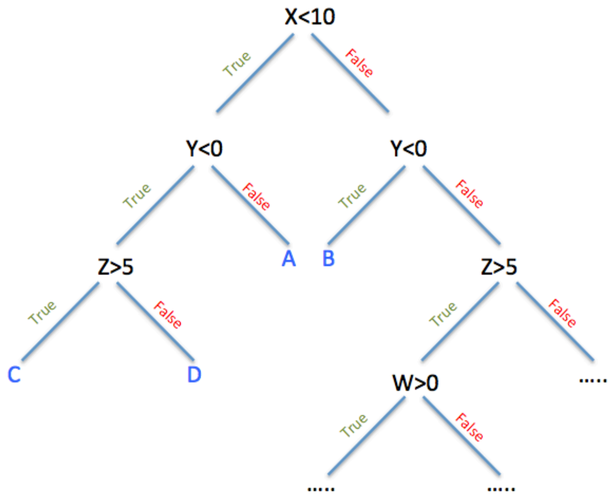
- separating hyperplanes with margin usually deliver a more ‘confident’ solution
- the optimization task has effective solution methods
- not robust to outliers (those that are close to the separation hyperplane)
- choosing the kernel is black magic; common sense doesn’t always work
- when there is no prior belief in linear separability of the classes, one has to tune parameters



Logical classifiers: decision trees

- What is decision tree?
- What Is split?
- How can we choose the split?
- Can we overfit (and how)?
- What is pruning?
- Do you know any tricks with categorical data?





Decision tree

- easy to interpret
- don’t have many assumptions on what the solution should look like
- overfit easily
- not that great for large dimensions

Machine learning models ensembles

- What is blending? Bagging? Boosting?
- What is the best algorithm to blend? Boost?
- Could you tell me about bias-variance decomposition?
- Why should we add randomisation into bagging/boosting?

Other stuff:

- **other ways to measure quality**, e.g., comparison with random predictions
- **feature selection (PMI, DIA, Chi-square, ...)**
- how to deal with **label-imbalanced** datasets
- **how to deal with small training data**
- **tuning hyperparameters** methods (grid search, random search, bayesian optimization, gradient-based optimization)

Important special cases:

Sentiment analysis: building ‘sentimental words’ vocabularies, e.g.

- semi-automatic (given initial sentimental seed words)

- custom vocabulary building, e.g. for specific domains

Topic classification:

- topic hierarchy building; the less supervision there is, the better

- dealing with the case where there is no true topic in label list yet

Refs:

- Yandex Data School course on machine learning + similar lecture notes: (http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf)
  https://yandexdataschool.ru/edu-process/courses/machine-learning
  Andrew Ng ? Coursera? Etc etc etc

- The Elements of Statistical Learning and other classical books on machine learning (classification is everywhere)

- Martin/Jurafsky, Chapters 6-7 in Ed. 3 4.

- Intro into IR (NB, kNN, Rocchio, SVM,…) https://nlp.stanford.edu/IR-book/

- Wikipedia

- CSC lectures, 2014 [Russian]