Lab 8. Ensemble learning

Intro to Machine Learning

Fall 2018, Innopolis University

Lecture recap

- Ensemble learning
- Bagging
- Random Forests
- Boosting
- Adaboost

Discussion

- How can we increase accuracy with ensemble learning?
- How can we reduce variance with ensemble learning?
- Discuss Bias-Variance Tradeoff.

Single classifier

$$a(x) = C(b(x))$$

$$b : X \to R$$

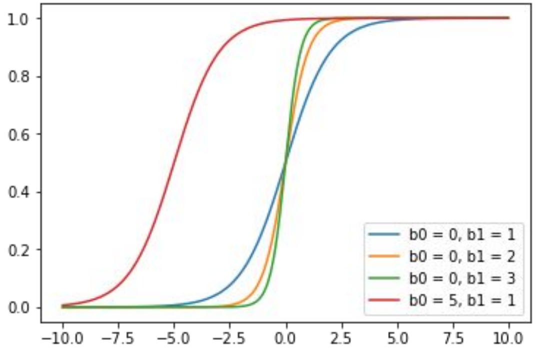
$$C : R \to Y$$

Single classifier

$$a(x) = C(b(x))$$

$$b : X \rightarrow R$$

$$C : R \rightarrow Y$$



$$b(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}$$

$$C(b(x)) = \begin{cases} 1 & b(x) > threshold \\ -1 & othwrwise \end{cases}$$

Ensemble Classifier

$$a(x) = C(F(b_1(x), ..., b_T(x)))$$
$$F : R^T \to R$$

Ensemble Classifier

$$a(x) = sign(\alpha_1 b_1(x) + ... + \alpha_T b_T(x))$$

- Train classifiers $b_i(x)$
- Train weights α_i

Bootstrap

From initial sample length ℓ we are creating random sampling with replacement with the same length ℓ.

Some objects are taken more, than once, some not included at all.

Part inside the new sample $1 - e^{-1} \approx 0.632$ if $\ell \to \infty$

```
a = range(10)
b = range(10, 20)

from sklearn.utils import resample
aa, bb = resample(a, b)

print(aa)
print(bb)
```

```
[9, 5, 1, 7, 3, 4, 7, 0, 3, 5]
[19, 15, 11, 17, 13, 14, 17, 10, 13, 15]
```

Bagging

1. Generate a bootstrap sample.
2. Randomly select the subset of the predictors.
3. Train the basic algorithm
4. Repeat *T* times from 1 to 3

$$a(x) = sign\left(b_1(x) + \ldots + b_T(x)\right))$$

Same weights for all classifiers

Exercise

Recall from lecture:

Now you are asked to illustrate this on generated data:

Why Bagging Works?

- Averaging reduces variance
- Let $Z_1, Z_2, \dots, Z_N$ be i.i.d random variables

$$Var\left(\frac{1}{N}\sum_i Z_i\right) = \frac{1}{N}Var(Z_i)$$

- generate N (say, 10) sets of samples drawing them from Normal distribution (choose different *mean*, but the same *std* for each set)

- print variance for each set

- calculate the average of samples over all sets and print the resulting variance, compare

Exercise. Hints

To sample from Normal distribution use:

```
samples = np.zeros((10, 100))
for i in range(10):
    samples[i, :] = np.array(np.random.normal(i, 0.5, 100))
```

At the end, you should get something similar to:

```
original variance for samples_0 0.2545698462514O566
original variance for samples_1 0.277644086043O631
original variance for samples_2 0.2702297276498457б
original variance for samples_3 0.2969463836б192955
original variance for samples_4 0.2385891842383342
original variance for samples_5 0.2824628301697803
original variance for samples_6 0.2310459357169371
original variance for samples_7 0.2747480815643359
original variance for samples_8 0.2057726595460977
original variance for samples_9 0.278433029938432
variance of the average 0.02310712303570283
```

Benefits from bagging

- Outliers not included into some samples.
- Variance is reduced.

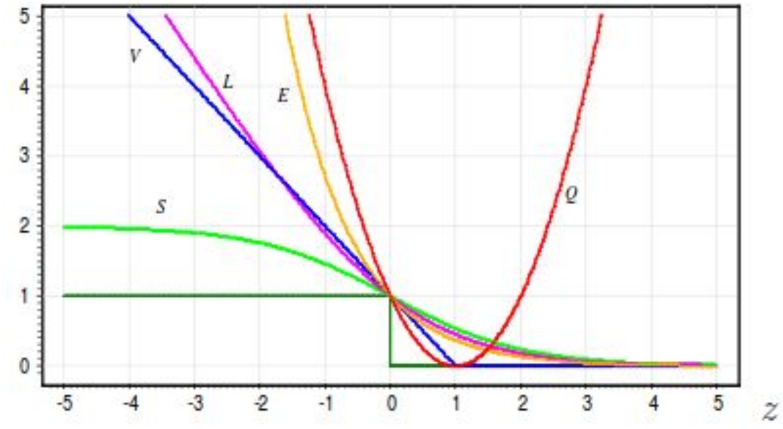
Loss Function

$$L_T = \sum_{i=1}^{l} \left[y_i \sum_{t=1}^{T} \alpha_t b_t(x_i) < 0 \right]$$

Heuristics

- Greedy algorithm to optimize the function. Adding $\alpha_t b_t(x)$ we assume $\alpha_1 b_1(x), ..., \alpha_{t-1} b_{t-1}(x)$ fixed. Optimizing only by $\alpha_t b_t(x)$ parameters.
- Change loss function to differentiable one.

Loss function



$$S(z) = 2(1 + e^{z}) - 1 \qquad \textit{sigmoid}$$

$$L(z) = \log_2(1 + e^{-z}) \qquad \textit{logarifmic}$$

$$V(z) = (1 - z)_+ \qquad \textit{piecewise linear}$$

$$E(z) = e^{-z} \qquad \textit{exponential}$$

$$Q(z) = (1 - z)^2 \qquad \textit{quadratic}$$

Adaboost

$$[y_i b(x_i) < 0] \leq e^{-y_i b(x_i)}$$

Adaboost

$$L_T \leq \tilde{L}_T = \sum_{i=1}^{l} exp(-y_i \sum_{t=1}^{T} \alpha_t b_t(x_i)) =$$

$$= \sum_{i=1}^{l} \underbrace{exp(-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i))}_{w_i} e^{-y_T \alpha_T b_T(x_i)}$$

Algorithm

Input: X^ℓ, Y^ℓ - training set. T - maximum number of base algorithms

Output: Base algorithms $b_t(x)$ and their weights α_t

1. Initialize all weights w_i = 1/ℓ for all i in 1,...,ℓ;
2. For all t in 1,...,T:
3. $$b_t = arg\min_b N(b, W^l)$$ $$N(b, W^l) = \sum_{i=1}^{l} w_i[b(x_i) = -y_i]$$ $$\alpha_t = \frac{1}{2}\ln\frac{1 - N(b, W^l)}{N(b, W^l)}$$
4. Recalculate weights $w_i = w_i exp(-\alpha_t y_i b_y(x))$
5. Normalize weights $w_0 = \sum_{i=1}^{l}; w_i = w_i / w_0.$

sklearn

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import AdaBoostClassifier

algo = LogisticRegression()
model = AdaBoostClassifier(base_estimator=algo, n_estimators=10)

model.estimator_weights_
model.estimators_[j]
```

Outliers

- High weights for misclassified objects.
- Adaboost tends to overfit to outliers.

Outliers

- We can filter outliers using adaboost.
- Objects with high weights could be removed as outliers.
- The model should be retrained.
- Adaboost could be used only for outliers filtering. Another algorithm could be used after filtering.

Discussion

What is the difference between **bagging** and **boosting**?

Homework (start on the lab)

- In this homework you are not provided with any template.
- You will not be given precise instructions.
    - Understand yourself, how you will measure the quality, decide what is an outlier and so on.
- You can use any methods from **sklearn**.
- **Report** is required and very important in this HW. The quality will be graded.
- You should think yourself, what should and what shouldn’t be in the report.
    - Describe your assumptions.
    - Analyze and explain results.
    - Justify your decisions.

Homework

1. We are going to predict region based on country data. For simplicity let’s make only two class classification - region is ‘EUROPE’ or not.
2. Download Countries dataset from Moodle.
3. Train adaboost classifier.
4. Find outliers in the dataset.
5. Retrain the model without outliers.

References

http://www.machinelearning.ru/wiki/images/0/0d/Voron-ML-Compositions.pdf

Boosting the margin: a new explanation for the effectiveness of voting methods / R. E. Schapire, Y. Freund, W. S. Lee, P. Bartlett //

https://web.stanford.edu/~hastie/Papers/samme.pdf

https://en.wikipedia.org/wiki/Bootstrapping_(statistics)