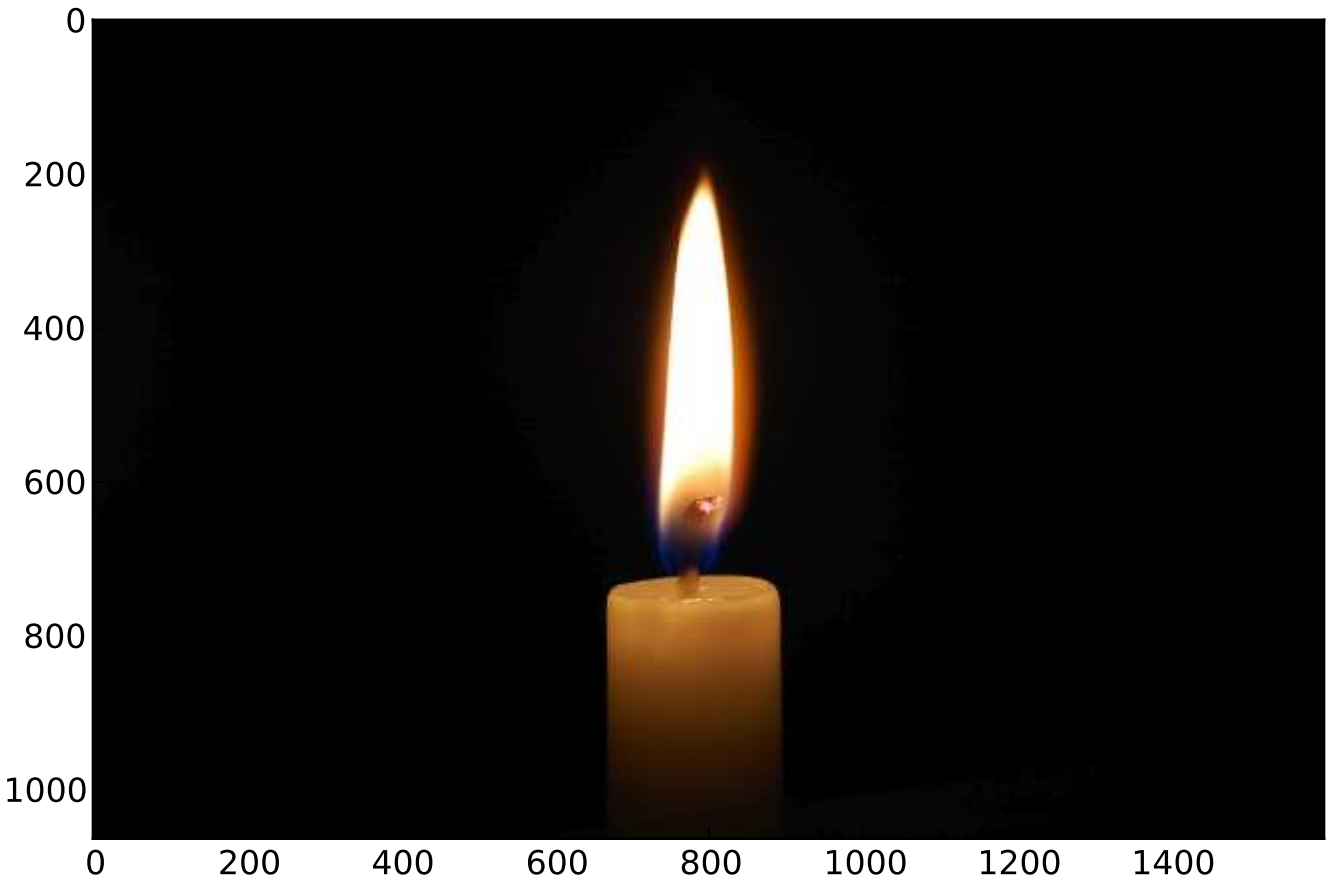
Semi-supervised learning in natural language processing

Anders Søgaard

Course outline

1. Supervised learning
2. Unsupervised learning and semi-supervised learning
3. **Learning from weighted data and transfer learning**
4. Applications to dependency parsing
5. Transfer learning in the blind


0
200
400
600
800
1000
0
200
400
600
800
1000
1200
1400

EM in scikits

```
weights=[1]*(X_labeled.shape[0])+[0]*U_size*2
y_train=list(y_labeled)+[1]*U_size+[0]*U_size
y_train=np.array(y_train)
scores=[clf.score(X_test,y_test)]
for _ in range(20): do
    for i in range(X_labeled.shape[0],X_train.shape[0]-U_size):  do
        if float(clf.predict(X_train[i,:]))==1:  then
            weights[i]=myMax(clf.predict_proba(X_train[i,:]))
            weights[i+U_size]=myMin(clf.predict_proba(X_train[i,:]))
        else
            weights[i]=myMin(clf.predict_proba(X_train[i,:]))
            weights[i+U_size]=myMax(clf.predict_proba(X_train[i,:]))
        end if
    end for
    clf.fit(X_train,y_train,sample_weight=weights)
    scores.append(clf.score(X_test,y_test))
end for
```

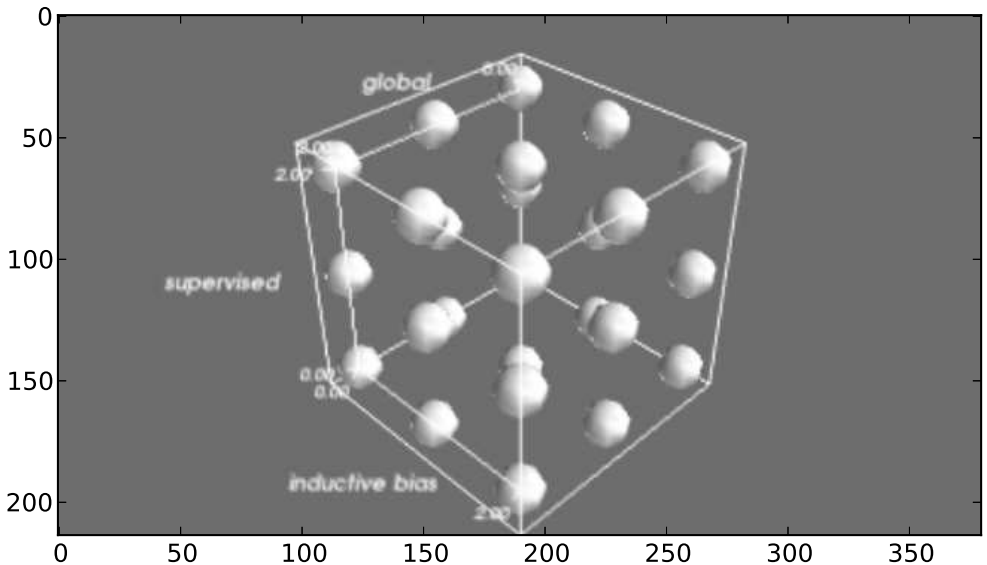
Data bias

Learning from weighted data

Importance weighting

Feature-based transfer learning

Main lessons from Monday and Tuesday



1. Labeled data is scarce and biased.
2. In semi-supervised learning, don't trust yourself.
3. Talk to other learners and ask them about their confidence (co-EM and tri-training).

Data bias

Learning under bias refers to learning where the assumption that data is identically distributed does not hold.

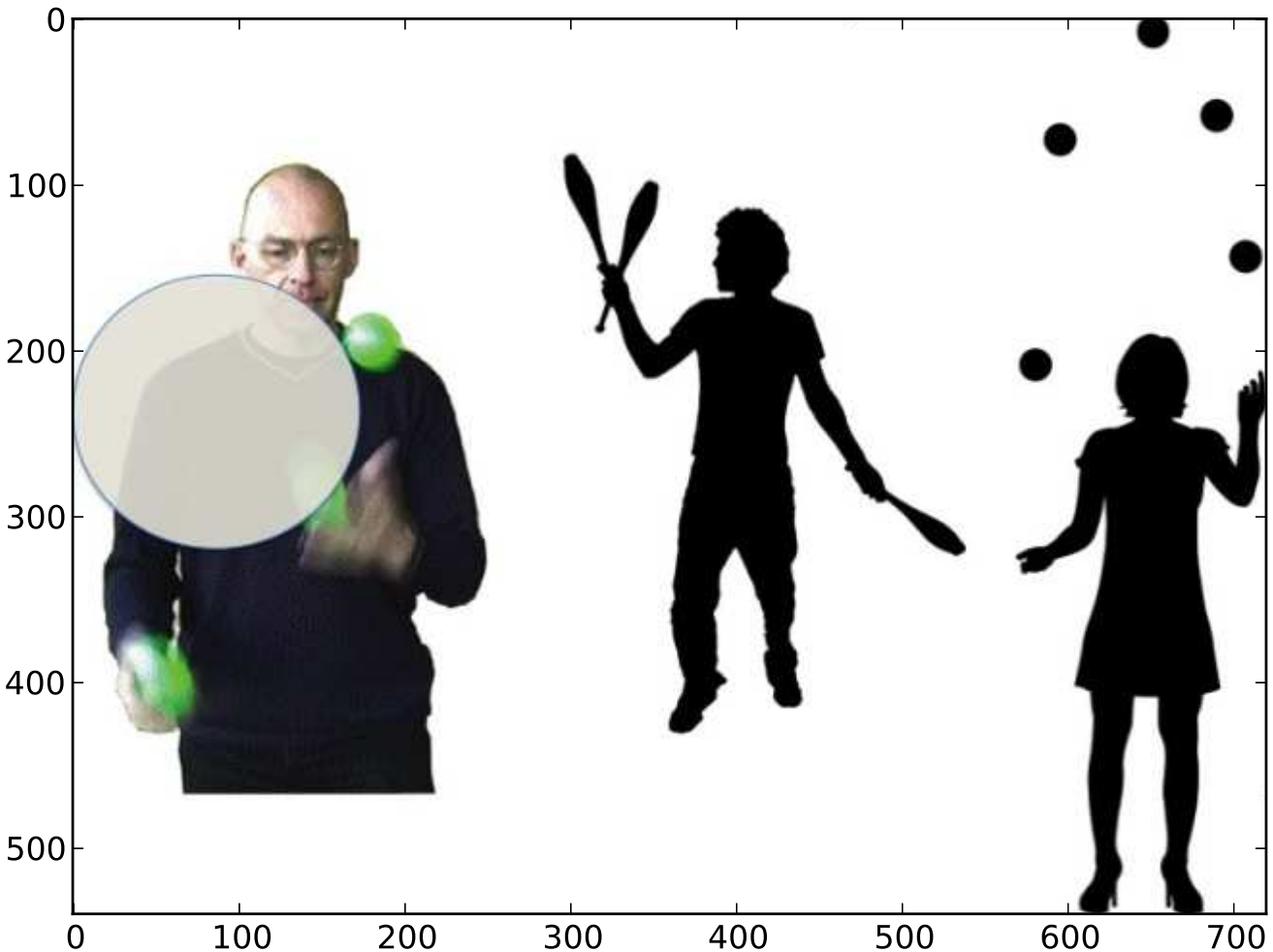
a) Sampling bias

b) Distribution bias

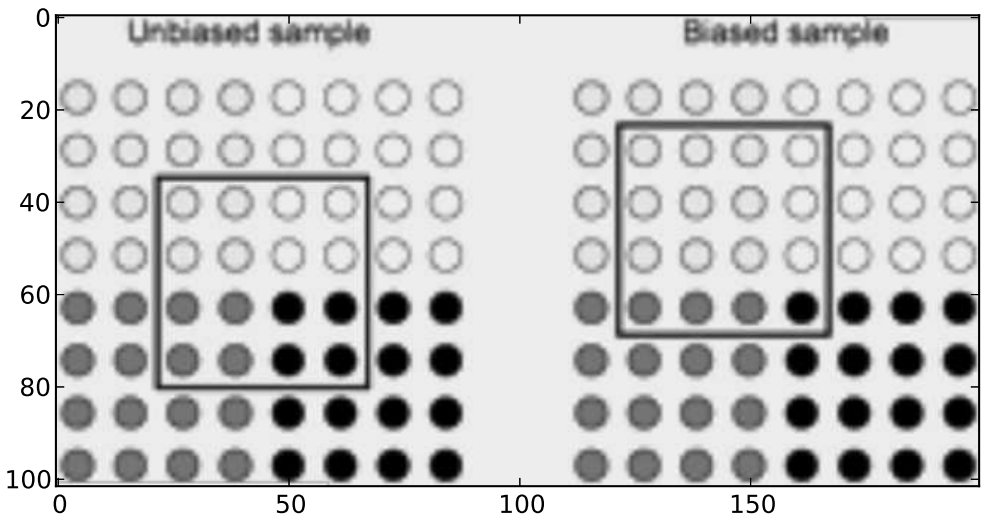
c) Problem bias

Important: While the domain adaptation literature often assumes a small sample of labeled target data, we *do not*.

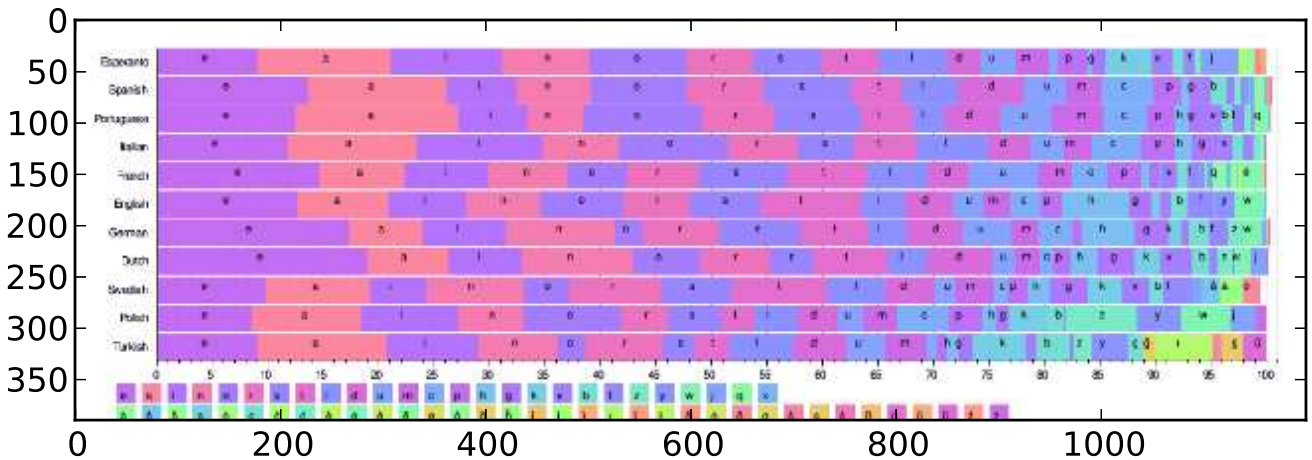
Data bias



Sampling bias



Distribution bias



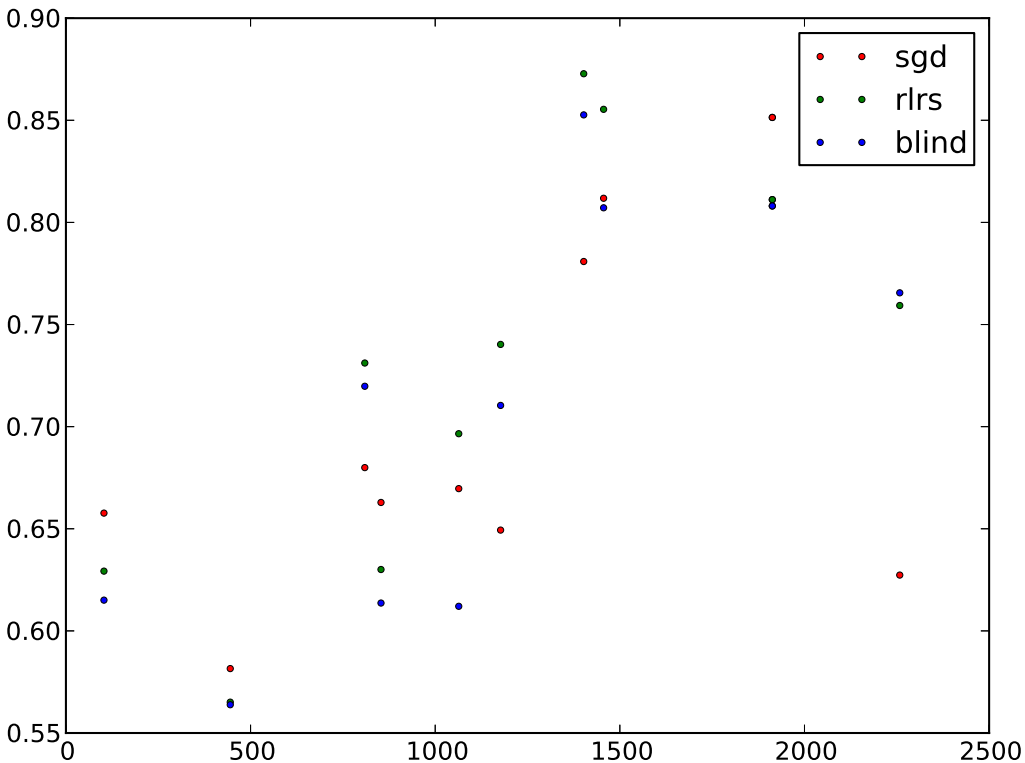
Data bias

	$P(X)$	$P(Y)$	$P(Y\|X)$
sampling bias	MAYBE	MAYBE	NO
distribution bias	MAYBE	MAYBE	MAYBE
problem bias	MAYBE	MAYBE	YES

- A bias in $P(X)$ has been referred to as *population drift* (Hand, 2006) or *instance adaptation* (Jiang and Zhai, 2007). *Covariate shift* (Shimodaira, 2001) only assumes a bias in $P(X)$.
- A bias in $P(Y)$ is typically referred to as *class imbalance*.
- A bias in $P(Y|X)$ is referred to as change in functional relation, *concept drift* (Hand, 2006) or *function adaptation* (Jiang and Zhai, 2007).

Exercise: Think of good NLP examples of bias in $P(X), P(Y), P(Y|X)$.

Evidence for bias in $P(X)$ in 20 Newsgroups



Note: Significant ρ also reported for Cora in Yang et al. (2012).

Exercises

1. Is domain adaptation a sampling bias correction problem or a distribution bias correction problem?
2. Can you distinguish the two looking only at data – and if so, how?

3. Can semi-supervised learning correct sampling bias, distribution bias, or both?

Note:

- We successfully applied semi-supervised learning to domain adaptation problems yesterday.
- The CoNLL 2007 and SANCL shared tasks on domain adaptation for parsing were both won by systems using semi-supervised learning.

Exercises

1. Is domain adaptation a sampling bias correction problem or a distribution bias correction problem?
2. Can you distinguish the two looking only at data – and if so, how?
   - A distribution bias may violate the smoothness assumption, even with a perfect feature model.
3. Can semi-supervised learning correct sampling bias, distribution bias, or both?

Note:

- We successfully applied semi-supervised learning to domain adaptation problems yesterday.
- The CoNLL 2007 and SANCL shared tasks on domain adaptation for parsing were both won by systems using semi-supervised learning.

Exercises

1. Is domain adaptation a sampling bias correction problem or a distribution bias correction problem?
2. Can you distinguish the two looking only at data – and if so, how?
   - A distribution bias may violate the smoothness assumption, even with a perfect feature model.
3. Can semi-supervised learning correct sampling bias, distribution bias, or both?
   - In both cases it depends on the expected error on the target distribution (which is unknown).

Note:

- We successfully applied semi-supervised learning to domain adaptation problems yesterday.
- The CoNLL 2007 and SANCL shared tasks on domain adaptation for parsing were both won by systems using semi-supervised learning.

Exercises

1. Is domain adaptation a sampling bias correction problem or a distribution bias correction problem?
2. Can you distinguish the two looking only at data – and if so, how?
   - A distribution bias may violate the smoothness assumption, even with a perfect feature model.
3. Can semi-supervised learning correct sampling bias, distribution bias, or both?
   - In both cases it depends on the expected error on the target distribution (which is unknown).

Note:

- We successfully applied semi-supervised learning to domain adaptation problems yesterday.
- The CoNLL 2007 and SANCL shared tasks on domain adaptation for parsing were both won by systems using semi-supervised learning.
- **. . . but it follows from the answer to 3) that semi-supervised learning only works if the bias is relatively small.**

Lessons

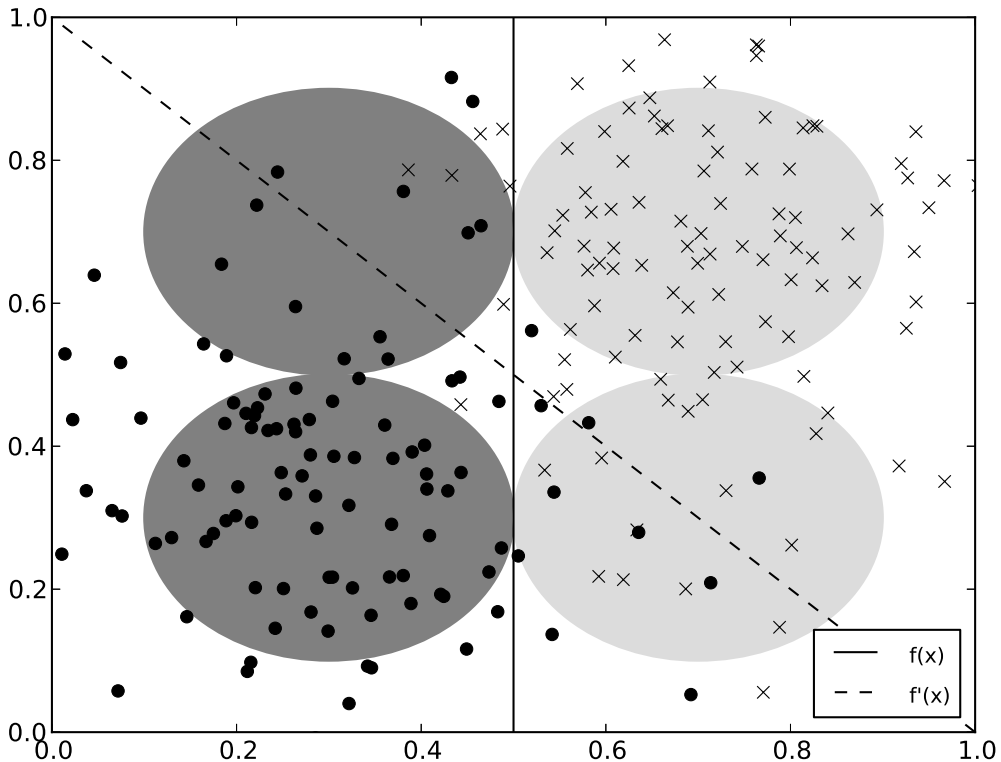
1. Labeled data is scarce and biased.
2. In semi-supervised learning, don't trust yourself.
3. Talk to other learners and ask them about their confidence (co-EM).
4. Global and unbiased methods are very sensitive to bias (KL-divergence).
5. Semi-supervised learning can correct bias when KL-divergence is *relatively* small.

Today: How can we go beyond semi-supervised learning to deal with biased data?

What can we afford to drop?

- **Datapoints?**
- Features?
- Parameters?

Motivation for weighted data



Weighted nearest neighbor

- In k-nearest neighbor it is easy to learn from weighted data.
- Simply let the k nearest neighbors vote on the class of the previously unseen instance and weight the vote of each neighbor by that data point's weight.
- Weighted k-nearest neighbor is currently not supported by SkLearn, but it is easy to modify the k-nearest neighbor version of the from-scratch code in Sect. 2.1 of the book to implemented weighted voting.
- Weighted k-nearest neighbor is often used in the nearest neighbor literature.

Weighted naive Bayes

Weighted naive Bayes (e.g. Zadrozny, 2004) *is* supported in SkLearn and works by weighting the MAP estimates. For illustration, consider the toy dataset from Monday reprinted here with weights in the left column:

β	y	zebra	viagra	venus
0.6	spam	0	1	0
0.2	non-spam	1	0	0
0.3	non-spam	0	0	1
	?	0	1	1

In the unweighted case the prior probability of observing spam was 1/3. In the weighted case $P(\text{spam}) = 0.6/1.1$. $P(\text{zebra} = 0 \mid \text{spam})$ is still 1, but while $P(\text{zebra} = 0 \mid \text{non-spam})$ was 1/2, it is now .3/.5.

Weighted perceptron

In weighted perceptron (e.g. Cavallanti et al., 2006), which is also supported in SkLearn, we make the learning rate dependent on the current instance, using the following update rule on $\mathbf{x}_n$:

$$\mathbf{w}^{i+1} \leftarrow \mathbf{w}^i + \beta_n \alpha(y_n - \text{sign}(\mathbf{w}^i \cdot \mathbf{x}_n))\mathbf{x}_n \quad (1)$$

Note: SCIKITS already implements weighted naive Bayes and weighted Perceptron using the flag sample_weight (but it is *not* mentioned in the documentation).

Weighted PA, MIRA and SVM

- The passive-agressive algorithm can be weighted by updating by a stepsize $\beta_n \alpha$ where β_n is the instance weight assigned to $\langle y_n, \mathbf{x}_n \rangle$.
- Søgaard and Haulrich (2011) also present an instance-weighted version of the MIRA algorithm and apply it to dependency parsing.
- Huang et al. (2007) present an instance-weighted learning algorithm for support vector machines. Here's the SVM objective with a capacity constant C to weight in-sample classification error:

$$\min_{\mathbf{w},b,\xi} C \sum_{i=1}^{N} \xi_i + \lambda ||\mathbf{w}||^2$$
$$\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \text{ for } i \in \{1, \ldots, N\} \quad (2)$$

The weighted objective, which is also supported in `scikits`:

$$\min_{\mathbf{w},b,\xi} C \sum_{i=1}^{N} \beta_i \xi_i + \lambda ||\mathbf{w}||^2$$
$$\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \text{ for } i \in \{1, \ldots, N\} \quad (3)$$

Note: Down-weighting can also be seen as encouraging under-fitting.

Importance weighting

- What weight function should we use in transfer learning?

Shimodaira (2001) proposes to use a maximum *weighted* (log-)likelihood estimate obtained by maximizing:

$$-\sum_{i=1}^{n} -w(\mathbf{x}) \log P(y_i|\mathbf{x}_i, \theta) \tag{4}$$

where $w(\cdot)$ is a function that assigns a weight to each of our labeled data points. Shimodaira (2001) also shows that the optimal weight function with sufficiently large samples is $P_T(\mathbf{x})/P_S(\mathbf{x})$, where $P_T(\mathbf{x})$ is the density function in the target domain, and $P_S(\mathbf{x})$ is the density function in the source domain.

- . . . but we can't compute density functions.

Importance weighting

You can obtain an estimated importance weight function by:

- domain classification (Zadrozny et al., 2004; Bickel and Scheffer, 2007; Søgaard and Haulrich, 2011),
- perplexity of target domain language model (Søgaard, 2011),
- compute reduced density functions (Søgaard and Plank, 2011),
- kernel mean matching (Huang et al., 2007), or
- minimizing KL-divergence (Sugiyama et al., 2007).

Robust weight functions:

- thresholding (Jiang and Zhai, 2007; Søgaard, 2011)
- binning in quantiles (Cortes et al., 2010)

Note: Jiang and Zhai (2007) weight by the error of a classifier trained on a small sample of target data, but we do not assume labeled target data.

Naive Bayes and importance weighting

- Bayesian learners are in theory *not* affected by sampling bias if we assume $P(y|\mathbf{x}, t = 1) = P(y|\mathbf{x})$ and $P(t = 1|\mathbf{x}, y) = P(t = 1|\mathbf{x})$.
- It follows that $P_s(y|\mathbf{x}) = P_t(y|\mathbf{x})$
- In practice, however, we have limited data, and a biased sample will lead to estimates of $P(y|\mathbf{x})$ with higher variance where $P(t = 1|\mathbf{x})$ is low.
- A Naive Bayes classifier turns out to be more sensitive to bias in the marginal distribution of data, since no independence can in general be assumed between x_i and y and t.

Importance weighted Naive Bayes in scikits

- clf=nb()
- clf.fit(X_dom,y_dom)
- weights=clf.predict_proba(X_train)
- weights=weights[:,1].reshape(X_train.shape[0])
- clf.fit(X_train,y_train)
- print clf.score(X_test,y_test)
- clf.fit(X_train,y_train,sample_weight=weights)
- print clf.score(X_test,y_test)

Comparison

Source	**Target**	NB	W-NB	Perc.	W-Perc.
Hockey-IBM	Baseball-Mac	94.76	95.5	86.32	89.64
Autos-Crypt	Motorcycles-Electronics	67.00	78.00	63.84	69.30
Guns-Space	MidEast-Medicine	67.62	69.17	63.08	66.32
Graphics-Misc(Politics)	Windows-Misc(Religion)	94.58	95.20	90.71	*88.24*

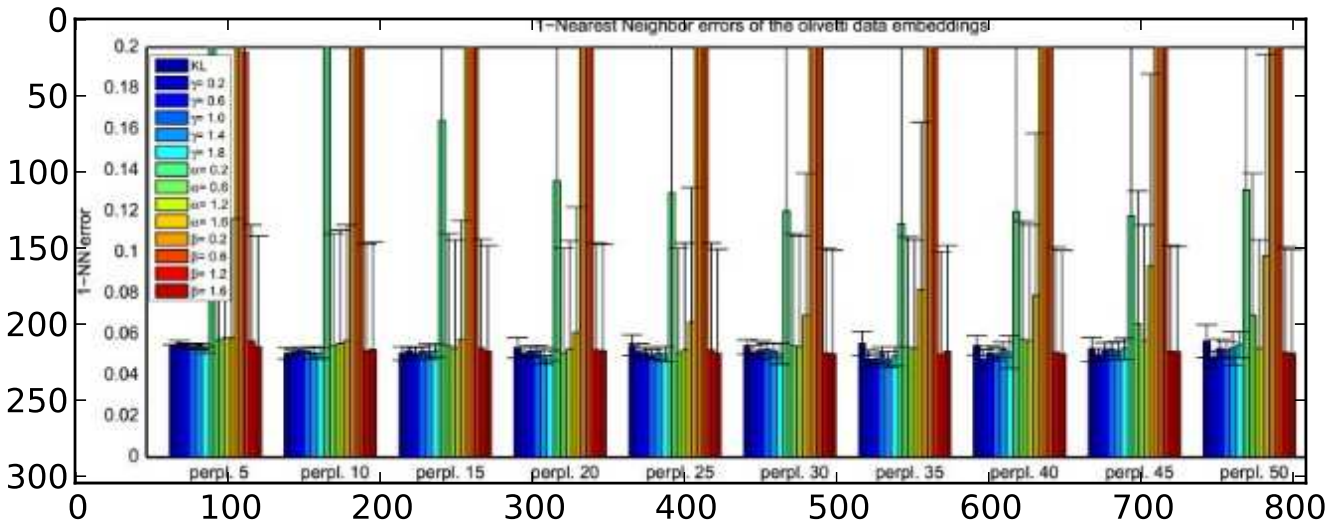
W-NB applied to WSD

acc(%)	cl	bl	sys	cl	bl	sys	cl	bl	sys	cl	bl	sys
art	7 †	15.5	21.1	5	45.1	36.6	3	46.5	53.5	2	81.7	93.0
authority	4	4.1	2.7	4	11.0	9.6	2	52.1	56.2	2	64.4	64.4
bar	7†	26.1	30.8	5	33.6	37.4	2	54.2	57.0	1	86.9	95.3
bum	2	11.1	13.9	2	19.4	11.1	2	38.9	41.7	2	86.1	91.7
chair	3	21.9	28.1	3	28.1	31.3	3	57.8	54.7	2	85.9	92.2
channel	5†	25.0	31.8	4	31.8	34.1	2	54.6	59.1	1	77.3	97.7
child	2	10.0	8.3	2	10.0	6.7	2	13.3	6.7	2	16.7	6.7
church	3	6.0	8.0	3	6.0	12.0	2	42.0	46.0	1	72.0	86.0
circuit	5†	20.0	28.0	3	26.7	37.3	2	42.7	42.7	1	88.9	97.3
day	2†	0	0	1	12.5	12.5	1	62.5	62.5	1	100	100
detention	3†	20.7	34.5	3†	41.4	44.8	3†	62.1	62.1	2	86.2	89.7
dyke	2	26.7	26.7	2	40.0	40.0	2	46.7	46.7	2	53.3	53.3
facility	2	48.2	50.0	2	50.0	53.6	2	69.6	67.9	1	92.9	98.2
fatigue	2	17.5	10.0	2†	32.5	32.5	1	60.0	60.0	1	90.0	97.5
feeling	2	8.2	2.0	2	20.4	30.6	1	53.1	59.2	1	77.6	91.8
grip	6†	7.5	5.0	6†	22.5	27.5	4†	55.0	57.5	3†	85.0	92.50
hearth	2	32.1	35.7	1	42.9	42.9	1	57.1	57.1	1	82.1	89.3
holiday	1	0	0	1	8.7	8.7	1	43.5	47.8	1	95.7	95.7
lady	1†	4.9	4.9	1†	12.2	9.8	1	7.3	4.9	1	12.2	2.4
material	5†	10.0	13.3	4†	11.7	13.3	2	53.3	56.7	1	91.7	98.3
mouth	4†	10.9	4.3	4	19.6	10.9	3	56.5	56.5	2	80.4	82.6
nation	3†	0	0	2	3.8	3.8	2	50.0	69.2	1	65.4	88.5
nature	5†	7.7	10.3	4†	25.6	23.1	2	69.2	71.8	1	87.2	97.4
post	4	5.3	12.3	3	21.1	21.1	2	63.2	66.7	1	77.2	89.5
restraint	6	7.3	7.3	3	29.3	29.3	2	65.9	68.3	1	78.0	87.8
sense	3	8.7	8.7	3	23.9	23.9	2	84.8	87.0	1	91.3	93.5
spade	3	25.0	32.1	3	25.0	25.0	2	57.1	60.7	2	75.0	82.1
stress	3†	3.1	12.5	3	28.1	28.1	2	65.6	81.3	1	87.5	96.9
yew	2	24.0	44.0	2	24.0	32.0	1	60.0	52.0	1	80.0	88.0
mic.av	-	14.9	17.6**	-	24.9	25.9	-	52.4	54.7**	-	77.2	84.0**
mac.av	-	15.9	18.8**	-	24.4	25.1	-	52.9	55.3**	-	77.2	83.8**

Robust importance weighting?

- Dasgupta and Long (2003) and Cortes et al. (2010) show that importance weighting can hurt in finite sample cases.
- The reason is that distributions must be *estimated*.
- Cortes et al. (2010) suggest to implement a bias-variance trade-off by binning weights in q quantiles.
- It is shown that for some q this gives robust importance weighting, but Cortes et al. (2010) suggest to estimate q on held-out data (which is not available in our case).
- **Technical note:** The problem is related to why *KL*-divergence is sometimes suboptimal. Hero et al. (2002) show that Renyi divergence with $\alpha = 0.5$ is optimal for relatively similar distributions: $D_\alpha(P, Q) = \frac{1}{\alpha - 1} \log(\sum_i \frac{P(x_i)^\alpha}{Q(x_i)^{\alpha-1}})$
- Thresholding (Jiang and Zhai, 2007) is an alternative.
- We report on dependency parsing experiments with binning and thresholding tomorrow.

Olivetti faces with $\alpha = 1.2$ optimal (Bunte et al., 2012)



What can we afford to drop?

- Datapoints?
- **Features?**
- Parameters?

Motivation for changing feature representations

When data is biased, e.g. in cross-domain sentiment analysis (Blitzer et al., 2007), some features transfer:

a) The cell phone is *horrible*.
b) The museum is *horrible*.

Some features are not transferred:

a) The cell phone is *handy*.
b) The museum is *handy*. (?)

Some features may even change polarity:

a) The cell phone is *small*.
b) The museum is *small*.

Structural correspondence learning - adapted

- Reduce the feature space to features that have support in T_S and T_T.
- Learn a perceptron $\mathbf{w}$ from T_S.
- Select k most predictive features f from $\mathbf{w}$ (pivot features).
- Train a classifier to predict the occurrence of each f in T_T.
- Construct $\mathbf{w}'$ from pivot features and predictive features g for each f with same sign as f's weight.

A simple alternative to SCL

- Satpal and Saragawi (2007) consider random subspaces to minimize divergence.
- Friday we will consider a similar approach but where no unlabeled target data is available.


RCA
STEREO
XYPBO-806
P-1975 Polar
Music AB
HASTA MAÑANA
(B. Andersson-S. Anderson-B. Ulvaeus)
ABBA
0
100
200
300
400
500
600