Machine Learning B (2025)
Home Assignment 1

Bar Segal xsb740

Contents

1 Numerical comparison of kl inequality with its relaxations and with Hoeffding's inequality (40 points) [Yevgeny]

Solution

We compare four one-sided upper confidence bounds on the true Bernoulli bias $p = P(X = 1)$ given an observed sample mean $\hat{p} = \frac{1}{n}\sum_{i=1}^{n} X_i$, at confidence $1-\delta$. Denote

$$\epsilon = \frac{\ln(1/\delta)}{n}.$$

1. **Hoeffding/Pinsker (closed form).** From Pinsker's relaxation of the KL inequality (Lemma 2.28) or directly from Hoeffding,

$$p \le \hat{p} + \sqrt{\frac{\ln(1/\delta)}{2n}}.$$

   Implementation (clipped at 1):

```
hoeff = np.clip(p_hat + np.sqrt(np.log(1/delta)/(2*n)), 0, 1)
```

2. **KL-inverse (tight bound).** Solve for $p \in [\hat{p}, 1]$:

$$\mathrm{kl}(\hat{p}\|p) = \hat{p}\ln\frac{\hat{p}}{p} + (1-\hat{p})\ln\frac{1-\hat{p}}{1-p} \le \epsilon$$

   by binary search.

```
def kl_div(q, p):
    # Bernoulli KL: KL(q||p)
    t1 = 0 if q==0 else q*np.log(q/p)
    t2 = 0 if q==1 else (1-q)*np.log((1-q)/(1-p))
    return t1 + t2

def kl_upper(phat, eps, tol=1e-9):
    # returns smallest p>=phat with kl_div(phat,p)<=eps
    lo, hi = phat, 1.0
    if kl_div(phat, hi) <= eps:
        return 1.0
    while hi - lo > tol:
        mid = 0.5*(lo+hi)
        if kl_div(phat, mid) > eps:
            hi = mid
        else:
            lo = mid
    return 0.5*(lo+hi)
```

3. **Pinsker's relaxation (same as Hoeffding).** From Lemma 2.28,

$$\mathrm{kl}(\hat{p}\|p) \geq 2(p-\hat{p})^2 \implies p \leq \hat{p} + \sqrt{\frac{\epsilon}{2}},$$

i.e. identical to the Hoeffding form above.

4. **Refined Pinsker (Corollary 2.32).** A sharper inversion gives

$$p \leq \hat{p} + \sqrt{2\,\hat{p}\,\epsilon + 2\,\epsilon},$$

clipped at 1. In code:

```
refined = p_hat + np.sqrt(2 * p_hat * epsilon) + 2 * epsilon
refined = np.clip(refined, 0, 1)
```

Plot and Zoomed in Plot

Evaluating each bound on a grid $\hat{p} = 0, 0.001, \ldots, 1$, with $n = 1000, \delta = 0.01$, then plotting:

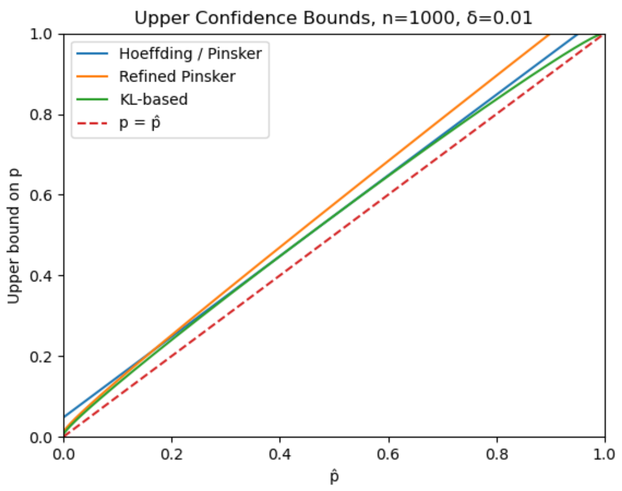

Figure 1: All four upper bounds versus $\hat{p}$.

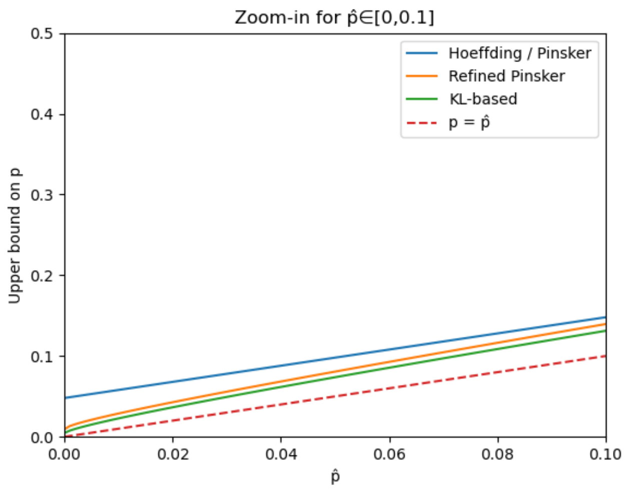

Figure 2: Zoom-in on $\hat{p} \in [0, 0.1]$.

Lower bounds

Comparing Hoeffding's lower bound on p with kl lower bound on p for the same values.

Reusing the KL–inverse for both bounds We write one divergence and two bisecting inverses, differing only in interval and endpoint update:

```
def kl_div(q, p):
    t1 = 0 if q==0 else q*np.log(q/p)
    t2 = 0 if q==1 else (1-q)*np.log((1-q)/(1-p))
    return t1 + t2

def kl_upper(q, eps, tol=1e-9):
    if q==1: return 1.0
    lo, hi = q, 1.0
    while hi-lo > tol:
        mid = (lo+hi)/2
        if kl_div(q, mid)>eps: hi = mid
        else:                  lo = mid
    return (lo+hi)/2

def kl_lower(q, eps, tol=1e-9):
    if q==0: return 0.0
    lo, hi = 0.0, q
    while hi-lo > tol:
        mid = (lo+hi)/2
        if kl_div(q, mid)>eps: lo = mid
        else:                  hi = mid
    return (lo+hi)/2

kl_up = np.array([kl_upper(q, epsilon) for q in p_hat])
```

```
kl_lo = np.array([kl_lower(q, epsilon) for q in p_hat])
```

Notice that between `kl_upper` and `kl_lower` the only changes are

- the initial interval: $[q, 1]$ vs. $[0, q]$, and
- which endpoint moves when $\mathrm{KL}(q\|\mathrm{mid}) > \varepsilon$.

No new search logic is needed—everything is shared.
And the resulting plot:

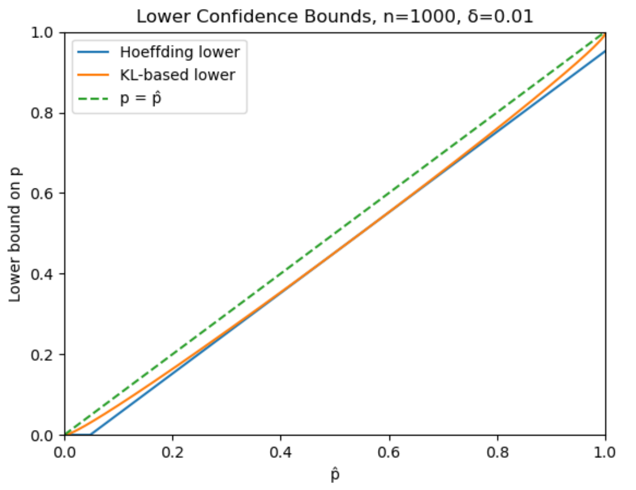

Figure 3: Hoeffding vs. KL lower-confidence bounds.

Conclusions

- Hoeffding/Pinsker gives the simplest closed-form but is not the tightest, especially in the tails, it is also shifted up a bit.
- Refined Pinsker improves at small or large $\hat{p}$ by using variance factors.
- The KL-inverse bound is the tightest everywhere, at the cost of a tiny binary search.
- For applications where $\hat{p}$ lies near 0 or 1 (rare events), the improvement can be dramatic.
- For task 4 we can see the straight Hoeffding line plunging to zero for small $\hat{p}$. A smoothly curved KL-inverse line, lying always above it and hugging the $45°$ diagonal more closely.

2 Occam's razor with kl inequality (30 points) [Yevgeny]

Solution

1. Proof of Theorem 3.38 (Occam's kl-razor inequality). We must show

$$\Pr\Big(\exists\, h \in \mathcal{H} :\ \mathrm{kl}\big(\hat{L}(h,S) \,\|\, L(h)\big) \ \geq\ \frac{\ln \frac{1}{\pi(h)\,\delta}}{n}\Big) \ \leq\ \delta.$$

Since $\sum_h \pi(h) \leq 1$, we apply the union bound with per-hypothesis confidence

$$\delta_h \ =\ \pi(h)\,\delta, \qquad \sum_h \delta_h = \delta.$$

For each fixed h, Theorem 2.27 (the one-sided KL-inequality) gives

$$\Pr\Big(\mathrm{kl}\big(\hat{L}(h,S)\|L(h)\big) \geq \tfrac{\ln \frac{1}{\delta_h}}{n}\Big) \ \leq\ \delta_h.$$

Summing over h yields the result. Note: it is crucial that $\pi(h)$ (and hence δ_h) be chosen before seeing the sample S, so that Theorem 2.27 applies independently to each hypothesis.

2. Proof of Corollary 3.39. Corollary 3.39 claims that under the same assumptions

$$\Pr\Big(\exists\, h :\ L(h) \geq \hat{L}(h,S) + \sqrt{\frac{2\,\hat{L}(h,S)\,\ln\big(1/(\pi(h)\delta)\big)}{n} + \frac{2\,\ln\big(1/(\pi(h)\delta)\big)}{n}}\Big) \ \leq\ \delta.$$

Starting from the refined Pinsker inversion (Corollary 2.32),

$$\mathrm{kl}(a\|b) \leq \epsilon \quad \Longrightarrow \quad b \ \leq\ a \ +\ \sqrt{2\,a\,\epsilon\ +\ 2\,\epsilon},$$

set

$$a = \hat{L}(h,S), \quad b = L(h), \quad \epsilon = \frac{\ln(1/(\pi(h)\,\delta))}{n}.$$

Whenever $\mathrm{kl}(\hat{L}(h)\|L(h)) < \epsilon$, the above gives the desired root-term bound on $L(h)$. By Theorem 3.38, the event "$\mathrm{kl}(\hat{L}\|L) \geq \epsilon$" happens with total probability at most δ, and its complement is exactly the square-root inequality, proving Corollary 3.39.

3. Comparison with Theorem 3.3 (classic Occam's razor). Theorem 3.3 (the Hoeffding-based Occam bound) states

$$\Pr\Big(\exists\, h :\ L(h) \geq \hat{L}(h,S) + \sqrt{\frac{\ln\big(1/(\pi(h)\delta)\big)}{2\,n}}\Big) \ \leq\ \delta.$$

Our KL-razor version replaces the constant Hoeffding-radius $\sqrt{\ln(1/(\pi(h)\delta))/(2n)}$ by the data-dependent

$$\sqrt{\frac{2\,\hat{L}(h,S)\,\ln(1/(\pi(h)\delta))}{n}+\frac{2\,\ln(1/(\pi(h)\delta))}{n}},$$

which is strictly smaller as soon as $\hat{L}(h,S)<\frac{1}{2}$. Thus the KL-based bound adapts to the observed empirical loss, yielding faster convergence for well-performing hypotheses, whereas the classic bound remains uniform but more conservative.

3 Numerical comparison of the kl and split-kl inequalities (30 points) [Yevgeny]

This implementation reuses the Bernoulli KL-divergence and its binary-search inversion to compute, for each parameter $p_{1/2}$, both the one-sided KL-inverse bound on $p-\hat{p}_n$ and the split-KL bound (by averaging two indicator-based inversions), then Monte Carlo–averages the results to smooth out noise, and finally plots the two curves. The resulting figure clearly shows that the split-KL bound is uniformly tighter than the standard KL-inverse bound across all values of $p_{1/2}$.

Python implementation

```
# 1) set up parameters for Ex.2.6
n2         = 100
delta2     = 0.05
eps_kl2    = np.log((n2)/delta2) / n2
eps_sp2    = np.log(2/delta2)    / n2

# 2) grid over p_{1/2} in [0,1]
grid = np.linspace(0,1,101)

# 3) number of MC replications for smoothing
reps = 200

# 4) containers
B_kl2     = np.zeros((grid.size, reps))
B_sp2     = np.zeros((grid.size, reps))

# 5) Monte Carlo loop
for i, p12 in enumerate(grid):
    # build ternary distribution
    p0 = p1 = (1-p12)/2
    probs = [p0, p12, p1]
    for j in range(reps):
        X = np.random.choice([0,0.5,1.0], size=n2, p=probs)
        phat = X.mean()
        # KL based on p phat
        pu = kl_upper(phat, eps_kl2)
```

```
            B_kl2[i,j] = pu - phat
            # split KL: two indicators
            ph1  = np.mean(X>=0.5)
            ph2  = np.mean(X>=1.0)
            r1   = kl_upper(ph1, eps_sp2)
            r2   = kl_upper(ph2, eps_sp2)
            B_sp2[i,j] = 0.5*(r1 + r2) - phat

# 6) average and clip
mean_kl2   = np.clip(B_kl2.mean(axis=1), 0, None)
mean_sp2   = np.clip(B_sp2.mean(axis=1), 0, None)

# 7) plot
plt.figure(figsize=(6,4))
plt.plot(grid, mean_kl2, label='KL based')
plt.plot(grid, mean_sp2, label='Split KL')
plt.axhline(0, color='gray', ls='--', lw=0.8)
plt.xlabel(r'$p_{1/2}$')
plt.ylabel(r'bound on $p-\hat p_n$')
plt.title(r'KL vs Split-KL, $n=100,\;\delta=0.05$')
plt.legend()
plt.grid(True)
plt.show()
```

And the resulting plot:

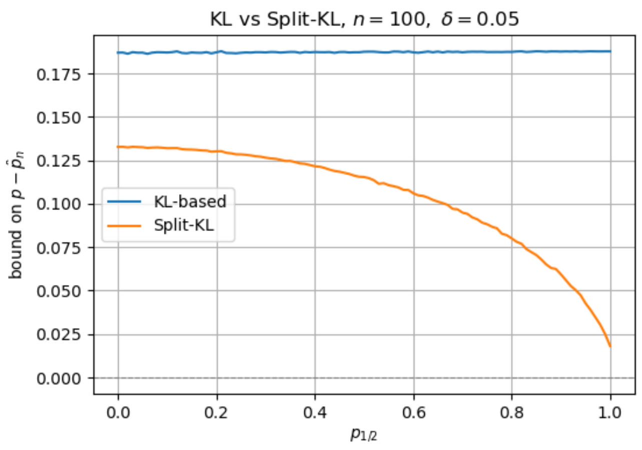

Figure 4: Hoeffding vs. KL lower-confidence bounds.

Reflection Because the split-KL uses only a $\ln(2/\delta)$ penalty rather than $\ln((n)/\delta)$, it consistently produces a *smaller* error bound across all values of $p_{1/2}$. In practice this means a tighter confidence guarantee for multi-level or composite alphabets.