به نام خدا

سامانه‌های یادگیری ماشین توزیع‌شده (پاییز ۱۴۰۴)

تمرین کامپیوتری ۱

مهلت ارسال: ۱۳ / ۰۸ / ۱۴۰۴

استاد درس: دکتر دوستی

دستیار طراح: علی خرم‌فر

قوانین و ملاحظات

نحوه ارسال تمرین:

- تمامی فایل‌ها باید در یک فایل فشرده با نام DMLS-CA1-StudentID ارسال شوند.
- کدهای مربوط به هر بخش را بر اساس جدول انتهای همین فایل ذخیره کرده و همراه گزارش ارسال کنید.
- تمامی کدهای ارسال‌شده باید امکان اجرای مجدد داشته باشند. اگر تنظیمات خاصی برای اجرا نیاز است، آن را ذکر کنید.
- کدهای ارسال‌شده باید توسط خودتان اجرا شده باشند و نتایج اجرا در فایل ارسالی مشخص باشد.

رعایت اصول آکادمیک و صداقت علمی:

- این تمرین باید به‌صورت **فردی** انجام شود. هرگونه همکاری یا نوشتن تمرین به‌صورت گروهی ممنوع است.
- در صورت مشاهده تشابه در پاسخ، تمامی افراد درگیر نمره صفر دریافت خواهند کرد و موضوع به استاد گزارش خواهد شد.

استفاده از ابزارهای هوش مصنوعی:

استفاده از ابزارهایی مانند ChatGPT، Gemini، Copilot و موارد مشابه مجاز است، اما تحت شرایط زیر:

- نحوه استفاده از این ابزارها را در گزارش خود توضیح دهید (ابزارهای استفاده‌شده، کاربردهای مشخص و موارد مرتبط).
- تمامی پرامپت‌ها و لینک‌های استفاده‌شده را در انتهای گزارش قرار دهید.
- عدم ارائه این اطلاعات به منزلهٔ سرقت علمی محسوب شده و منجر به نمره صفر خواهد شد.

مهلت ارسال و جریمه تأخیر:

- امکان ارسال تمرین با تأخیر تا ۲ روز وجود دارد. برای هر روز تأخیر ۱۰ درصد جریمه اعمال می‌شود.
- تأخیر به‌صورت ساعتی محاسبه شده و پس از دو روز تأخیر، تمرین پذیرفته نخواهد شد.

ارزیابی حضوری:

- ارزیابی تمرین به‌صورت حضوری انجام خواهد شد.
- محل ارزیابی: آزمایشگاه NLP، طبقه منفی یک، دانشکده مهندسی برق و کامپیوتر ساختمان شماره ۲.

توضیحات تمرین

لطفاً پیش از شروع کار بر روی تمرین، به نکات زیر توجه فرمایید.

- حتماً ویدئوی راه‌اندازی کلاستر را به‌دقت مشاهده کنید و مطمئن شوید به کلاستر درس دسترسی دارید.
- آدرس‌های کلاسترهای درس به شرح ذیل است:

```
dml0:  172.18.32.200
dml1:  172.18.32.201
dml2:  172.18.32.202
dml3:  172.18.32.203
```

- برای راحتی در توسعه و تست کد، از ماشین مجازی لینوکسی خود استفاده نمایید تا ترافیک کلاستر (به‌خصوص در ساعات آخر مهلت تمرین) افزایش نیابد. پس از اطمینان از عملکرد کد، می‌توانید آن را روی کلاستر اجرا کنید.
- در صورت بروز مشکل با ایمیل زیر در ارتباط باشید:

```
khoramfar@ut.ac.ir
```

سؤال ۱. محاسبه عدد π با مسئله بازل و چالش توزیع بار

۴۰ نمره

در قرن هجدهم، یکی از مشهورترین مسائل حل‌نشده در دنیای ریاضیات، پیدا کردن مقدار دقیق مجموع معکوس مجذورات اعداد طبیعی بود. ریاضیدانان بزرگی تلاش کردند اما موفق نشدند، تا اینکه در سال ۱۷۳۴، لئونارد اویلر پاسخی برای آن کشف کرد:

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \cdots = \frac{\pi^2}{6}$$

این نتیجه، ارتباطی عمیق بین اعداد صحیح و عدد π را آشکار ساخت. در این تمرین، از این رابطه برای تخمین عددی π با محاسبه N جمله اول سری استفاده می‌شود.

$$\pi \approx \sqrt{6 \times \sum_{k=1}^{N} \frac{1}{k^2}}$$

برای شبیه‌سازی یک سناریوی واقعی‌تر که در آن هزینه محاسباتی وظایف یکسان نیست، فرض می‌شود هزینه محاسبه هر جمله از سری با شماره آن جمله (k) افزایش می‌یابد. این وابستگی با تابع زیر مدل می‌شود که باید در پیاده‌سازی خود از آن استفاده کنید.

```python
def compute_term(k):
    s = 0
    for i in range(k // 1000):
        s += i * i
    return 1.0 / (k * k)
```

تابع محاسبه هر جمله با هزینه محاسباتی وابسته به k

Problem Parameters:

- Number of series terms (N): 500,000

۱.۱ پیاده‌سازی سریال

برنامه‌ای بنویسید که N جمله سری را با استفاده از تابع `compute_term(k)` به صورت سریالی محاسبه و جمع کند. در نهایت مقدار π را تخمین زده و چاپ نمایید. این کد را بر روی **۱ نود و ۱ هسته** اجرا کنید.

۲.۱ پیاده‌سازی موازی - روش اول

الگوریتم بخش ۱.۱ را با استفاده از mpi4py موازی کنید. بازه [1, N] را بین فرآیندها به صورت بازه‌های متوالی و هم‌اندازه تقسیم کنید. به عنوان مثال، اگر N=1000 و ۴ فرآیند داشته باشیم، فرآیند ۰ بازه [1, 250]، فرآیند ۱ بازه [251, 500] و به همین ترتیب را محاسبه می‌کند. این پیاده‌سازی را در دو پیکربندی زیر اجرا نمایید:

- **پیکربندی ۱:** ۱ نود و ۲ هسته.
- **پیکربندی ۲:** ۲ نود، هر کدام ۲ هسته.

۳.۱ پیاده‌سازی موازی - روش دوم

الگوریتم بخش ۱.۱ را مجدداً به صورت موازی پیاده‌سازی کنید، اما این بار N جمله را به صورت دوره‌ای بین فرآیندها توزیع کنید. به عنوان مثال، اگر ۴ فرآیند داشته باشیم، فرآیند ۰ جملات ... ,1, 5, 9، فرآیند ۱ جملات ... ,2, 6, 10 و به همین ترتیب را محاسبه می‌کند. این پیاده‌سازی را نیز در دو پیکربندی زیر اجرا نمایید:

- **پیکربندی ۱:** ۱ نود و ۲ هسته.
- **پیکربندی ۲:** ۲ نود، هر کدام ۲ هسته.

۴.۱ تحلیل و گزارش نهایی

نتایج تمام بخش‌ها را به همراه زمان اجرا در هر هسته و زمان اجرای کل گزارش دهید. عملکرد روش‌های مختلف (سریال و دو روش موازی) را در تمام پیکربندی‌ها با یکدیگر مقایسه و تحلیل نمایید. برای درک و نمایش بهتر نتایج، از نمودار استفاده نمایید.

سؤال ۲. رگرسیون لجستیک موازی با الهام از یادگیری فدرال

۵۰ نمره

یکی از پارادایم‌های نوین در یادگیری ماشین توزیع‌شده، **یادگیری فدرال (Federated Learning)** است. این رویکرد به چندین کلاینت (مانند دستگاه‌های موبایل یا بیمارستان‌ها) اجازه می‌دهد تا بدون به اشتراک گذاشتن داده‌های خصوصی خود، در آموزش یک مدل مشترک که توسط یک سرور مرکزی مدیریت می‌شود، همکاری کنند.

یکی از چالش‌های اساسی در یادگیری فدرال، ناهمگونی داده‌ها (Data Heterogeneity) است. برخلاف یادگیری ماشین متمرکز که در آن داده‌ها معمولاً از یک توزیع یکسان و مستقل (IID) نمونه‌برداری می‌شوند، در سناریوی فدرال، داده‌های هر کلاینت ممکن است توزیع آماری متفاوتی داشته باشند. در این سؤال، ما این چالش را با فراهم کردن سه مجموعه داده مجزا برای کلاینت‌ها شبیه‌سازی می‌کنیم.

در این شبیه‌سازی، فرآیند با rank 0 نقش سرور مرکزی و سایر فرآیندها نقش کلاینت‌ها را ایفا می‌کنند. ما از الگوریتمی الهام گرفته از میانگین‌گیری فدرال (Federated Averaging یا FedAvg) برای آموزش یک طبقه‌بند رگرسیون لجستیک استفاده خواهیم کرد. در این روش، هر کلاینت مدل را برای چند مرحله به صورت محلی آموزش می‌دهد و سپس سرور با میانگین‌گیری مدل‌های دریافتی، مدل سراسری را به‌روز می‌کند.

۱.۲ پیاده‌سازی سریال به عنوان معیار پایه (Baseline)

ابتدا یک طبقه‌بند رگرسیون لجستیک استاندارد را به صورت سریال پیاده‌سازی کنید. هر سه مجموعه داده پیوست‌شده به تمرین را بارگیری کرده و آن‌ها را با یکدیگر ادغام کنید تا یک مجموعه داده کلی به دست آید. سپس این مجموعه داده کلی را به دو بخش آموزش (٪۸۰) و آزمون (٪۲۰) تقسیم نمایید. مدل خود را با استفاده از پارامترهای زیر روی داده‌های آموزش اجرا کرده و دقت نهایی آن را بر روی داده‌های آزمون به همراه زمان کل آموزش گزارش دهید.

- **تعداد ایپاک‌ها:** ۳۰
- **نرخ یادگیری:** ۰.۰۱
- **مقداردهی اولیه وزن‌ها:** بردار صفر

۲.۲ پیاده‌سازی موازی و تحلیل سناریوهای مختلف

برنامه‌ای با استفاده از mpi4py بنویسید که الگوی سرور/کلاینت را پیاده‌سازی کند. در این پیاده‌سازی، فرآیند rank 0 سرور و فرآیندهای rank 1, 2, 3 کلاینت‌ها هستند. هر کلاینت یکی از مجموعه داده‌های مجزا را بارگیری می‌کند (rank 1 داده اول، rank 2 داده دوم و الی آخر) و ٪۸۰ آن را برای آموزش و ٪۲۰ را برای آزمون محلی کنار می‌گذارد.

پارامترها:

- **مقداردهی اولیه وزن‌ها:** مدل سراسری در سرور باید با **بردار صفر** آغاز شود.
- **نرخ یادگیری :** ۰.۰۱

الگوریتم کلی در هر دور ارتباطی (Communication Round) به شرح زیر است:

۱. **سرور (rank 0):** وزن‌های مدل سراسری فعلی را به تمام کلاینت‌ها ارسال می‌کند.

۲. **کلاینت‌ها (rank > 0):** وزن‌های دریافتی را به عنوان نقطه شروع مدل محلی خود قرار می‌دهند و آن را برای چند ایپاک محلی (Local Epoch) روی داده‌های محلی خود آموزش می‌دهند.

۳. **کلاینت‌ها:** پس از اتمام ایپاک‌های محلی، وزن‌های مدل آموزش‌دیده خود را به سرور مرکزی ارسال می‌کنند.

۴. **سرور:** منتظر می‌ماند تا وزن‌ها را از تمام کلاینت‌ها دریافت کند. سپس این وزن‌ها را میانگین‌گیری کرده تا مدل سراسری جدیدی برای دور بعدی بسازد.

این پیاده‌سازی را در پیکربندی **۲ نود، هر کدام ۲ هسته** (مجموعاً ۴ فرآیند: ۱ سرور و ۳ کلاینت) برای دو سناریوی زیر اجرا کنید:

- **سناریوی اول:** ۱ دور ارتباطی و ۱۰ ایپاک محلی.
- **سناریوی دوم:** ۱۰ دور ارتباطی و ۱ ایپاک محلی.

ارزیابی: پس از اتمام تمام دورهای ارتباطی، سرور باید مدل سراسری نهایی را به تمام کلاینت‌ها ارسال کند. هر کلاینت دقت این مدل نهایی را بر روی داده‌های آزمون محلی خود می‌سنجد. میانگین این دقت‌ها به عنوان دقت نهایی سیستم گزارش شود. برای هر دو سناریو، زمان اجرای آموزش، زمان اجرای کل، دقت هر کلاینت و میانگین دقت نهایی را گزارش دهید. برای درک و تفسیر بهتر نتایج توزیع داده‌های آموزشی را بررسی کنید.

۳.۲ تحلیل اثر کلاینت مخرب و مقایسه سناریوها - ۱۰ نمره امتیازی

- **آزمایش کلاینت مخرب:** سناریوی دوم (۱۰ دور ارتباطی، ۱ اپیاک محلی) را مجدداً اجرا کنید، اما این بار یکی از کلاینت‌ها (به‌دلخواه) نرخ یادگیری خود را به 0.5 تغییر دهد، در حالی که سایر کلاینت‌ها روی نرخ یادگیری 0.01 باقی می‌مانند. زمان اجرا و میانگین دقت نهایی را برای این حالت نیز گزارش دهید.
- در گزارش نهایی خود به موارد زیر پاسخ دهید:

۱. عملکرد دو سناریوی اول و دوم را از نظر دقت مدل‌ها و زمان آموزش با یکدیگر مقایسه کنید. trade-off بین افزایش محاسبات محلی (اپیاک‌های بیشتر) و افزایش ارتباطات (دورهای بیشتر) را شرح دهید.

۲. نتایج آزمایش کلاینت مخرب را تحلیل کنید. نرخ یادگیری بالا چه تأثیری بر دقت مدل سراسری نهایی گذاشت؟ دلیل این پدیده را توضیح دهید.

سؤال ۳. بنچمارک پیاده‌سازی‌های مختلف BLAS در NumPy — ۲۰ نمره

در این سوال قصد بنچمارک کردن پیاده‌سازی‌های مختلف BLAS در کتابخانه‌ی numpy را داریم. برای نصب کتابخانه‌های شتاب‌دهنده‌ی جبر خطی مختلف به همراه numpy، می‌توانید راهنمای نصب numpy را مطالعه نمایید. در این سوال از لپتاپ یا کامپیوتر شخصی خود استفاده کرده و برای پردازنده‌های x86 (اینتل یا AMD) می‌توانید از pip و MKL استفاده کنید. برای دستگاه‌های Apple نیز از pip و کتابخانه accelerate استفاده نمایید. برای بنچمارک کردن این کتابخانه‌ها از محاسبات زیر استفاده کنید:

۱.۳ محاسبه دترمینان ماتریس مربعی

۲.۳ محاسبه معکوس ماتریس مربعی

برای هر دو بخش از دو ماتریس با ابعاد ۱۰۰۰۰ و ۲۰۰۰۰ استفاده کنید. (اگر سیستم شما حافظه کافی ندارد، می‌توانید از ابعاد کوچکتر مانند ۵۰۰۰ و ۸۰۰۰ استفاده کنید). همچنین برای بدست آوردن زمان مصرفی از کتابخانه `time` در پایتون استفاده نمایید.

نحوه تحویل پروژه

۱. گزارش پروژه را در قالب فایل PDF تهیه کنید.

۲. تمام فایل‌های مربوط به تمرین را طبق ساختار جدول زیر در پوشه‌های مشخص‌شده قرار داده و در انتها با فرمت zip در سامانه elearn بارگذاری کنید.

۳. اسکریپت‌های اجرای بر روی بیش از یک هسته باید در فایل‌هایی با پسوند sh. نوشته شوند.

سؤال	بخش	نام فایل‌ها
۱	۱.۱ (سریال)	`pi_serial.py`

	۲.۱ (موازی - روش اول)	`pi_parallel_v1.py`
		`pi_parallel_v1_c1.sh`
		`pi_parallel_v1_c2.sh`

	۳.۱ (موازی - روش دوم)	`pi_parallel_v2.py`
		`pi_parallel_v2_c1.sh`
		`pi_parallel_v2_c2.sh`

۲	۱.۲ (سریال - Baseline)	`logreg_serial.py`

	۲.۲ و ۳.۲ (موازی)	`logreg_fedavg.py`
		`scenario_1.sh`
		`scenario_2.sh`
		`malicious.py`
		`malicious.sh`

۳	۱.۳ (بنچمارک دترمینان)	`bench_det_10k.py` `bench_det_20k.py`
	۲.۳ (بنچمارک معکوس)	`bench_inv_10k.py` `bench_inv_20k.py`

توزیع نمرات

سؤال	بخش	نمره	توضیح
۱ (۴۰ نمره)	۱.۱ (سریال)	۵	پیاده‌سازی صحیح نسخه سریال
	۲.۱ (موازی - روش اول)	۱۰	پیاده‌سازی موازی با تقسیم متوالی
	۳.۱ (موازی - روش دوم)	۱۰	پیاده‌سازی موازی با تقسیم دوره‌ای
	۴.۱ (گزارش و تحلیل)	۱۵	مقایسه عملکرد، تحلیل توازن بار و نمودارها
۲ (۵۰ نمره)	۱.۲ (سریال - Baseline)	۱۰	پیاده‌سازی صحیح رگرسیون لجستیک سریال
	۲.۲ (پیاده‌سازی موازی)	۳۰	پیاده‌سازی الگوریتم FedAvg و اجرای صحیح سناریوهای اول و دوم
	۳.۲ (بخش امتیازی)	۱۰	اجرای آزمایش کلاینت مخرب
۳ (۲۰ نمره)	۱.۳ (بنچمارک دترمینان)	۵	اجرای بنچمارک برای دترمینان ماتریس
	۲.۳ (بنچمارک معکوس)	۵	اجرای بنچمارک برای معکوس ماتریس
	گزارش و تحلیل	۱۰	تحلیل نتایج، مقایسه کتابخانه‌ها و گزارش نهایی