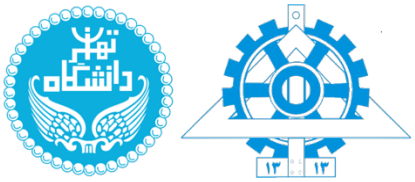
به نام خدا



سامانه‌های یادگیری ماشین توزیع‌شده (پاییز ۱۴۰۳)

تمرین کامپیوتری ۱

موعد تحویل: ۱۴۰۳/۷/۳۰

لطفا پیش از شروع کار بر روی تمارین، به نکات زیر توجه فرمایید.

- حتما ویدئوی راه اندازی کلاستر را به دقت مشاهده کنید و مطمئن شوید به کلاستر درس دسترسی دارید.
- توجه داشته باشید که آدرس‌های استفاده شده در ویدئو مربوط به گذشته هستند و باید از آدرس های جدید استفاده کنید. آدرس‌های جدید 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 لینوکس خود استفاده نمایید تا ترافیک کلاستر به (خصوص درساعات آخر مهلت تمرین) افزایش نیابد. پس از اطمینان از عملکرد کد، میتوانید آن را روی کلاستر اجرا کنید.
- توصیه میشود از زبان Python به همراه با کتابخانه py4mpi برای کدنویسی استفاده کنید. در آدرس زیر، راهنمای استفاده از این کتابخانه آمده است:

Tutorial — MPI for Python documentation (mpi4py.readthedocs.io)

- حتی‌المقدور سوالات خود را در گروه تلگرام درس مطرح نمایید. در غیر این صورت از طریق ایمیل ar.amir23@yahoo.com با حل تمرین مربوط در ارتباط باشید.
- به هیچ وجه کد یا پاسخ سوالات را به اشتراک نگذارید. هرگونه تقلب به منزله نمره صفر برای طرفین است.
- میزان تاخیر، تا دو روز مجاز است. تاخیر به صورت ساعتی محاسبه شده و هر روز ۱۰ درصد نمره کم میشود. تحویل تمرین پس از دو روز تاخیر امکان پذیر نخواهد بود.

---

۱. در این سوال قصد داریم تا با استفاده از سری تیلور عدد $\sqrt{2}$ را محاسبه کنیم. به منظور ساده‌سازی محاسبات، پس از اعمال تبدیل اویلر به سری زیر می‌رسیم :

$$\sqrt{2} = \sum_{k=0}^{\infty} \frac{(2k+1)!}{2^{3k+1}(k!)^2} = \frac{1}{2} + \frac{3}{8} + \frac{15}{64} + \frac{35}{256} + \frac{315}{4096} + \frac{693}{16384} + \cdots$$

الف) سری فوق را بدون هرگونه بهینه‌سازی پیاده کنید و برای ۵۰۰۰ جمله بر روی یک نود[۱] و یک هسته اجرا نمایید.

ب) با بررسی فرمول بالا مشاهده می شود که هم در صورت و هم در مخرج کسر بسیاری از محاسبات انجام شده با مراحل قبل همپوشانی دارند و امکان استفاده از نتایج مرحله قبل وجود دارد. به عنوان مثال، برای محاسبه $k!$ کافی‌است k را در $(k-1)!$ ضرب کرد. با در نظر گرفتن این موضوع الگوریتم مرحله **الف** را بهینه‌سازی کنید و برای ۵۰۰۰ جمله بر روی یک نود و یک هسته اجرا نمایید. زمان اجرای الگوریتم نباید بیش از ۸ ثانیه باشد.

ج) الگوریتم مرحله **الف** را با پیکربندی‌های زیر برای ۵۰۰۰ جمله اجرا نمایید. برای توزیع بار محاسباتی از روش قطعه‌بندی متوالی استفاده کنید. به عنوان مثال با در نظر گرفتن ۲ هسته و ۵۰۰۰ جمله، هسته شماره ۱ جمله ۱ تا ۲۵۰۰ و هسته شماره دو جملات ۲۵۰۱ تا ۵۰۰۰ را محاسبه می‌کند.

۱) یک نود و دو هسته

۲) دو نود هر کدام دو هسته

د) الگوریتم مرحله **ب** را با پیکربندی‌های زیر اجرا نمایی . برای توزیع بار محاسبات از روش بخش **ج** استفاده کنید.

۱) یک نود و دو هسته

۲) دو نود و هر کدام دو هسته

ه) در بخش **ج** عدم توازن در توزیع بار محاسباتی بین نودها، منجر به افزایش زمان اجرا می‌گردد. الگوریتم بخش **ج** را مجددا اجرا کنید اما این بار، محاسبات را به شکلی توزیع کنید که زمان اجرا کاهش یابد و بر روی دو نود هر کدام دو هسته اجرا نمایید.

نتیجه هر مرحله از آزمایش را به همراه زمان اجرا گزارش کرده و نتایج قسمت‌های مختلف را مقایسه کرده و تحلیل نمایید.

۲. یکی از کاربردهای الگوریتم‌های توزیع شده، حفظ حریم شخصی کاربران است. برای پیاده‌سازی این الگوریتم‌ها ، باید در نظر گرفت که هر نود داده‌های مخصوص به خود را دارد و تمام یا بخش‌هایی از الگوریتم را خودش اجرا می‌کند و تنها برای تجمیع نتایج، نودها با یکدیگر ارتباط برقرار می‌کنند. به طور کلی، برای تقسیم داده‌ها بین نودها دو روش تقسیم‌بندی افقی و عمودی مطرح می‌شود. در تقسیم‌بندی افقی، هر نود تعدادی از رکوردهای داده را می‌گیرد و هر رکورد تمام ویژگی‌هایش را دارد. در حالی که در تقسیم‌بندی عمودی، تقسیم‌بندی بر اساس ویژگی‌های داده‌ها انجام میشود و هر نود تعدادی از ویژگی‌ها را برای تمام رکوردها دارد (Aggarwal CC, 2008).

[۱] node

در این سوال قصد داریم الگوریتم طبقه‌بندی Logistic Regression را به صورت توزیع شده پیاده‌سازی کنیم. برای رعایت حفظ حریم شخصی، قصد داریم از روش تقسیم‌بندی افقی استفاده نماییم. به این منظور، الگوریتم طبقه‌بندی را به صورت زیر تغییر می‌دهیم:

- تمامی نودها وزن‌های اولیه را با عدد صفر مقداردهی می‌کنند .
- در هر مرحله از به روزرسانی وزن‌ها، هر نود با استفاده از داده‌های خود اقدام به محاسبه مشتق‌های جزئی مربوط به هر وزن می‌کند.
- سپس مشتق‌های به دست آمده را با نود اصلی به اشتراک گذاشته و نود اصلی اقدام به بروز رسانی وزن ها می‌کند.
- در انتها نود اصلی وزن های به روز رسانی شده را برای تمام نود های دیگر ارسال کرده و آن ها نیز اقدام به به روز رسانی وزن های خود می‌کنند.
- برای این سوال مجموعه داده data.npy به همراه برچسب های مربوطه فراهم شده است. با توجه به صورت مسئله داده‌ها را بین نودهای موجود تقسیم کنید. در هر نود ۲۰٪ از داده‌ها را برای مرحله آزمایش جدا کنید.

الف) الگوریتم Logistic regression را به صورت سریال پیاده سازی کرده، اجرا کنید و مقدار دقت را گزارش نمایید.

ب) با استفاده از کتابخانه mpi4py پیاده سازی مرحله **الف** را بر روی یک نود و دو هسته اجرا کنید.

ج) پیاده سازی مرحله **ب** را بر روی دو نود و دو هسته اجرا نمایید.

نتیجه هر مرحله از آزمایش را به همراه زمان اجرا گزارش کرده و نتایج قسمت های مختلف را مقایسه کرده و تحلیل نمایید. توجه نمایید که میزان دقت باید در تمامی نود ها محاسبه گردد.

۳. در این سوال قصد بنچمارک کردن پیاده سازی‌های 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 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:

الف) محاسبه ضرب دو ماتریس مربعی با ابعاد یکسان

ب) محاسبه معکوس ماتریس مربعی

برای هر دو بخش از دو ماتریس با ابعاد ۱۰۰۰۰ و ۲۰۰۰۰ استفاده کنید. همچنین برای بدست آوردن زمان مصرفی از کتابخانه time در پایتون استفاده نمایید.

نحوه تحویل پروژه

۱- گزارش pdf

۲- تمرین انجام شده را با ساختار زیر پوشه‌بندی و نام‌گذاری کنید. در نهایت به صورت فایل zip در سامانه elearn بارگذاری نمایید.

۳- کدهای bash مربوط به اجرا بر روی بیش از یک هسته که با استفاده از slurm صورت می‌گیرد در فایل‌ها با پسوند sh نوشته شود.

سوال	بخش	نام فایل ها
۱	الف	sqrt_a.py
	ب	sqrt_b.py
	ج	sqrt_c_1.py sqrt_c_1.sh sqrt_c_2.py sqrt_c_2.sh
	د	sqrt_d_1.py sqrt_d_1.sh sqrt_d_2.py sqrt_d_2.sh
	ه	sqrt_e.py sqrt_e.sh
۲	الف	LogReg_a.py LogReg_a.sh
	ب	LogReg_b.py LogReg_b.py
	ج	LogReg_c.py LogReg_c.py
۳	الف	Bench_a_10k.py Bench_a_20k.py
	ب	Bench_b_10k.py Bench_b_20k.py

نحوه توزیع نمرات

سوال	بخش	نمره
۱ (۵۰ نمره)	الف	۵
	ب	۵
	ث	۱۰
	د	۱۰
	ج	۵
	گزارش	۱۵
۱ (۵۰ نمره)	الف	۱۰
	ب	۱۰
	ث	۱۰
	گزارش	۲۰
۱ (۲۰ نمره)	الف	۶
	ب	۶
	گزارش	۸

منبع

Aggarwal CC, P. S. (2008). Privacy-preserving data mining: models and algorithms. *Springer Science and Business Media*, 28.