به نام خدا

هوش محاسباتی: تمرین شبیه‌سازی سری دوم تاریخ تحویل: ۱۷ آذر ۱۴۰۲

* سوال اول و دوم را می‌توانید در محیط پایتون نیز پیاده‌سازی کنید.

* سوال سوم امتیازی است.

۱- فایل Ex1.mat شامل سه بردار نرخ سوخت، سرعت و میزان اکسید نیتروژن خروجی در یک اتومبیل است. می‌خواهیم نرخ سوخت را بر حسب میزان اکسید نیتروژن خروجی و سرعت تخمین بزنیم.

الف) با استفاده از دستور scatter3، متغیر خروجی را بر حسب دو متغیر ورودی رسم کنید.

ب) ۷۰۰ داده اول را به عنوان داده‌های آموزشی در نظر گرفته و از بقیه داده‌ها به عنوان داده‌های اعتبارسنجی (validation) استفاده کنید (داده‌ها به صورت تصادفی در بردار قرار داده شده‌اند).

ج) با استفاده از رگرسیون خطی، یک مدل خطی بر داده‌های آموزشی برازش (fit) کرده و خطای میانگین مربعات (Mean Square Error) را بر روی داده‌های آموزشی و اعتبارسنجی محاسبه کنید.

د) با استفاده از رگرسیون لاجیستیک (logistic) یک مدل بر داده‌های آموزشی برازش کرده و خطای میانگین مربعات را بر روی داده‌های آموزشی و اعتبارسنجی محاسبه کنید.

ه) با استفاده از دستور fitnet یا تولباکس nnstart، یک شبکه عصبی MLP با یک لایه پنهان بر داده‌های آموزشی برازش کرده و خطای میانگین مربعات را بر روی داده‌های آموزشی و اعتبارسنجی محاسبه کنید. تعداد نورون‌های لایه پنهان را به گونه‌ای تعیین کنید که کمترین خطا بر روی داده‌های اعتبارسنجی به دست آید.

$$MSE = \frac{1}{N}\Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad \hat{y}_i: Estimated\ Value \qquad y_i: Real\ Value \ ,$$

توجه: موارد زیر را در یک پوشه به نام Ex1 ذخیره نمایید:

- یک فایل pdf شامل توضیحات در مورد هر یک از بخش‌های (الف) تا (ه) و شکل‌ها و نتایج به دست آمده در هر قسمت. هم‌چنین با توجه به نتایج به دست آمده در مورد مزایا و معایب هر یک از سه طراحی (ج)، (د) و (ه) بحث کنید.
- کد MATLAB نهایی

۲- داده‌های سه بعدی که متعلق به دو کلاس ۰ و ۱ هستند به عنوان داده‌های آموزشی داده شده‌اند:

(ذخیره شده در فایل Ex2.mat)

هر ستون از ماتریس TrainData به ابعاد 90×4، شامل یک داده آموزشی سه‌بعدی و برچسب (کلاس) متناظر آن است (سه سطر اول داده ورودی بوده و سطر آخر برچسب متناظر با داده است).

ماتریس TestData به ابعاد 90×3 نیز به عنوان ۹۰ داده آزمون داده شده است.

با استفاده از شبکه‌های عصبی پرسپترون (در دو حالت زیر) و با استفاده از داده‌های آموزشی TrainData یک طبقه‌بندی‌کننده طراحی کنید. سپس کلاس متناظر با هر یک از داده‌های آزمون را تعیین کنید. (می‌توانید از Toolboxهای nnstart و nntool و توابع feedforwardnet و patternnet استفاده کنید)

الف) شبکه شامل یک لایه پنهان بوده و خروجی فقط شامل یک نورون باشد که صفر و یک بودن آن (یا ۱ و ۱-) کلاس متناظر را تعیین کند.

ب) شبکه شامل یک لایه پنهان بوده و دو نورون در خروجی باشد که هر کدام میزان عضویت در هر کلاس را تعیین کند. یعنی اگر برای یک داده، خروجی نورون اول ۰/۷ و خروجی نورون دوم ۰/۳ شد، یعنی داده مورد بررسی متعلق به کلاس اول است.

راهنمایی: ابتدا برای اینکه درک بهتری از پخش داده‌های دو کلاس در فضای سه‌بعدی داشته باشید، داده‌های دو کلاس را با دو رنگ متفاوت در فضای سه‌بعدی رسم کنید. ۲۰٪ از داده‌ها را به صورت تصادفی به عنوان داده اعتبارسنجی (validation) جدا کرده و در هر یک از دو حالت (الف) و (ب)، شبکه‌های مختلف با تغییر تعداد نورون‌های لایه پنهان ایجاد کرده و صحت طبقه‌بندی را بر روی داده‌های اعتبارسنجی محاسبه کنید. در هر دو حالت، شبکه‌ای را انتخاب کنید که بهترین نتیجه را بر روی داده‌های اعتبارسنجی ایجاد می‌کند و آن را بر روی داده‌های آزمون اعمال کنید.

توجه: موارد زیر را در یک پوشه به نام Ex2 ذخیره نمایید:

- یک فایل pdf شامل توضیحات در مورد طراحی شبکه عصبی، مثلاً تعداد لایه‌ها، تعداد نورون‌ها در هر لایه، تابع فعال‌سازی نورون‌ها و ... . هم‌چنین با توجه به نتایج به دست آمده در مورد مزایا و معایب هر یک از دو طراحی (الف) و (ب) بحث کنید.
- کد MATLAB نهایی برای هر یک از دو حالت (الف) و (ب)
- برچسب به دست آمده برای داده‌های آزمون با استفاده از هر یک از دو حالت (الف) و (ب) به صورت دو فایل Testlabel_a.mat و Testlabel_b.mat

* ۳- کد MLP_Example.ipynb به شما داده شده است. این کد یک پیاده‌سازی ساده برای یک شبکه MLP با الگوریتم ErrorBackPropagation را به شما نشان می‌دهد. برای راحتی، کد تقریباً تکمیل است و فقط چند بخش کوچک آن با ؟؟؟؟ مشخص شده است که شما بایستی آنها را تکمیل کنید.

الف) در این کد می‌توان از توابع فعالسازی tanh و logistic استفاده کرد. برای پیاده‌سازی الگوریتم ErrorBackPropagation به تعریف این توابع و مشتق آنها نیاز داریم. در بالای کد تعریف خود توابع آمده است. تعریف مشتق توابع را تکمیل کنید.

ب) در main، تعریف ورودی و خروجی مطلوب شبکه به صورت ناقص آمده است:

```
X = np.array(?????????????)
y = np.array(?????????????)
```

با در نظر گرفتن تابع XOR، تعریف دو متغیر X و y را تکمیل کنید (X ماتریسی به ابعاد 2 × 4 و y برداری سطری به طول ٤ است).

ج) تمام بخش‌های الگوریتم به جز بخش BackPropagation تکمیل هستند:

```
for i in range(len(self.weights)):
        layer = np.atleast_2d(a[i])
        delta = np.atleast_2d(deltas[i])
        self.weights[i] +=??????????????????
```

ابتدا همه بخش‌های کد را (شامل تعریف شبکه، لایه‌ها، خطا، خروجی هر لایه، مقادیر دلتا و ...) مطالعه نمایید. سپس با بررسی مفهوم متغیرهای *layer* و *delta* و با استفاده از متغیر *learning_rate* خط آخر را تکمیل کنید.

د) الگوریتم را با مقادیر *default* داده‌شده اجرا کرده و خروجی را بررسی کنید. آیا خروجی به‌دست‌آمده با خروجی مطلوب یکی است؟ توضیح دهید.

ه) در این کد، برخلاف الگوریتم توضیح‌داده‌شده در کلاس، در هر *epoch*، فقط یکی از الگوهای آموزشی به صورت تصادفی انتخاب شده و بر اساس آن، وزن‌ها به‌روز می‌شوند. بعد از خط مربوط به تعریف خطا:

```
error = y[i] – a[-1]
```

با استفاده از دستور *print*، هر ۵۰۰ *epoch* خطا را پرینت کنید یا نمودار تغییرات خطا را بر حسب شماره *epoch*ها رسم کنید. در مورد تغییرات خطا چه می‌توان گفت؟ آیا روند نزولی دارد؟ چرا؟

و) کد را برای دو تابع منطقی دلخواه دیگر (دو ورودی، یک خروجی) نیز اجرا کرده و خروجی‌ها را بررسی کنید. در مورد هر تابع چندین بار کد را اجرا کنید. آیا خروجی همیشه ثابت است؟ آیا خروجی همیشه با خروجی مطلوب یکسان است؟

ز) کد را برای *XOR* و دو تابع منطقی استفاده‌شده در بخش (و) با در نظر گرفتن تابع فعالسازی *logistic* اجرا کرده و خروجی‌ها را بررسی کنید. در مورد هر تابع چندین بار کد را اجرا کنید. آیا خروجی همیشه ثابت است؟ آیا خروجی همیشه با خروجی مطلوب یکسان است؟ در صورت لزوم، کد را برای به دست آوردن خروجی مطلوب تصحیح کنید.

توجه: موارد زیر را در یک پوشه به نام Ex3 ذخیره نمایید:

- کد تکمیل‌شده
- یک فایل pdf شامل توضیحات لازم و پاسخ به سوالات مطرح‌شده