

به نام کیمیاگر عالم



دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

یادگیری ماشین کاربردی

عنوان

تمرین سوم (HW03)

مدرس

دکتر احسان ناظر فرد

دانشجو

امیرحسین بابائیان

۴۰۱۱۳۱۰۰۲

ترم بهار ۰۲-۰۱

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)

فهرست

فهرست	۲
سوال اول	۵
بخش a	۵
بخش b	۵
بخش c	۵
بخش d	۶
بخش e	۶
سوال دوم	۶
بخش a : Pre-processing	۷
بخش b : Split the data	۷
بخش c : Run logistic regression	۷
بخش d : accuracy and confusion matrix	۷
بخش e : best probability threshold	۸
بخش f	۸
ذخیره سازی مدل	۸
استفاده از مدل ذخیره شده	۸
سوال سوم	۹
بخش a	۹
خواندن داده ها	۹
حذف ستون بدون کاربرد	۹
حذف سطرهاى خالى و كامنت هاى خالى	۹

۹.....	نصب hazm و nltk روی colab
۱۰.....	مصور سازی داده ها بر اساس لیبل
۱۰.....	عملیات tokenizer و normalizer
۱۰.....	عملیات tokenizer روی متن
۱۰.....	حذف ستون های اضافه
۱۱.....	دانلود stopwords فارسی و حذف کلمات
۱۱.....	لیبل انکودینگ
۱۱.....	تفکیک داده تست و آموزش
۱۱.....	استفاده از CountVectorizer و TFIDFVectorizer
۱۱.....	حذف محتویات اضافه در متن ها
۱۱.....	بخش b
۱۱.....	توضیحات
۱۲.....	پیاده سازی تابع fit
۱۲.....	پیاده سازی تابع predict
۱۲.....	استفاده از GNB
۱۳.....	بخش c
۱۳.....	استفاده از NB داده شده
۱۳.....	SKlearn در GNB
۱۴.....	سوال چهارم
۱۴.....	بخش a
۱۴.....	بخش b
۱۴.....	بخش c
۱۴.....	بخش d
۱۴.....	انتخاب مدل

- اسکیل کردن ۱۵
- ساخت مدل و ارزیابی اولیه ۱۵
- پیدا کردن همسایه ۱۵
- بخش e ۱۵
- نمونه اول ۱۵
- نمونه دوم ۱۶
- نمونه سوم ۱۶
- نمونه چهارم ۱۶
- نمونه پنجم ۱۶

سوال اول

بخش a

مدل رگرسیون لجستیک قابلیت پشتیبانی از چند کلاس را به صورت مستقیم دارد، بدون اینکه نیاز به آموزش و ترکیب چندین دسته بند دودویی داشته باشد. این کار با استفاده از تابع softmax در مدل رگرسیون لجستیک امکان پذیر است. تابع softmax برای محاسبه احتمال تعلق هر نمونه به هر یک از کلاس ها استفاده می شود. در این حالت، تعداد خروجی های مدل برابر با تعداد کلاس هاست و احتمالات تعلق به هر کلاس در خروجی مدل قرار می گیرد.

در مدل رگرسیون لجستیک، هدف پیش بینی احتمال برچسب دو کلاس ۰ و ۱ برای هر نمونه است. اما در مسئله تشخیص چند کلاس، ما باید احتمال تعلق هر نمونه به تمامی کلاس ها را پیش بینی کنیم. برای حل این مسئله، می توان از تابع softmax استفاده کرد.

تابع softmax، یک تابع محاسبه احتمالات مخصوص به مدل های چند کلاسه است که احتمال تعلق هر نمونه به هر یک از کلاس ها را محاسبه می کند. برای مثال، اگر مسئله شامل ۳ کلاس باشد، تابع softmax احتمال تعلق هر نمونه به هر یک از این ۳ کلاس را محاسبه می کند و خروجی مدل ۳ عدد احتمالی است که مجموع آن ها برابر با ۱ است.

در مدل رگرسیون لجستیک با استفاده از تابع softmax، می توان تعداد دلخواهی از کلاس ها را پشتیبانی کرد و احتمال تعلق هر نمونه به هر کلاس را برای تمامی کلاس ها به صورت همزمان محاسبه کرد، بدون نیاز به آموزش و ترکیب چندین دسته بند دودویی.

بخش b

Odds ration یک معیار آماری است که در رگرسیون لجستیک برای مقایسه شانس وقوع یک رویداد بین دو گروه استفاده می شود، به عبارت دیگر نسبت شانس نشان می دهد که احتمال وقوع یک رویداد نسبت به رویداد دیگر به چه شکلی است، اگر بیشتر از ۱ باشد یعنی احتمال گروه اول بیشتر است و اگر کمتر از صفر باشد یعنی احتمال گروه دوم بیشتر است و اگر صفر باشد به معنی یکسان بودن شانس است. این نسبت می تواند برای تخمین اثر پیش بینی کننده های مختلف بر نتیجه مورد علاقه در مدل های رگرسیون لجستیک استفاده شود.

بخش c

در طبقه بندی بیزی، احتمال وقوع یک خروجی یا نتیجه مشخص، با توجه به دسته های موجود و مشاهداتی که در هر دسته وجود دارد، محاسبه می شود. این روش برای پیش بینی و طبقه بندی در بسیاری از حوزه ها مفید است، اما معمولاً فقط با داده هایی که دارای ویژگی های دسته ای هستند به خوبی کار می کند، به عبارت دیگر، داده هایی که متغیرهای کیفیتی (categorical) دارند.

اما اگر داده‌های ما شامل ویژگی‌های عددی هستند، چگونه می‌توانیم از روش طبقه‌بندی بیزی استفاده کنیم؟ در این صورت، باید ویژگی‌های عددی را به دسته‌هایی تقسیم بندی کنیم. روش معمول برای انجام این کار، استفاده از روش تجزیه و تحلیل بندبند یا binning است. در این روش، با تقسیم داده‌های عددی به چند دسته با ویژگی‌های مشابه، عمل تبدیل دسته‌ای انجام می‌شود که به آن دسته بندی می‌گویند. سپس با استفاده از روش طبقه بندی بیزی می‌توان پیش‌بینی برای دسته‌های مختلف انجام داد.

در مجموع، با تبدیل ویژگی‌های عددی به دسته‌های مشابه و استفاده از روش طبقه بندی بیزی، می‌توان از این روش برای پیش‌بینی و طبقه بندی داده‌های عددی نیز استفاده کرد.

بخش d

به طور کلی می‌توان گفت در صورتی که از متد لاپلاس برای حذف صفر شدن احتمال استفاده کنیم، می‌توانیم احتمال هر حالتی را محاسبه نموده و به نوعی داده‌ی جدید نیز تولید کنیم، از این رو به با دیتای کم هم می‌تواند پاسخ گو باشد.

بخش e

بله، رگرسیون لجستیک می‌تواند مرز تصمیم غیرخطی را با استفاده از تبدیلات غیرخطی و ویژگی‌های ترکیبی ایجاد کند. به عنوان مثال، با افزودن ویژگی‌هایی که حاصل ضرب دو یا چند ویژگی قبلی هستند، می‌توان مرز تصمیمی غیرخطی را به دست آورد. همچنین با استفاده از تبدیلات غیرخطی مانند تابع لگاریتمی، تابع توانی یا توابع تک‌متغیره غیرخطی، می‌توان مرز تصمیمی غیرخطی را به دست آورد. بنابراین، با استفاده از تبدیلات مناسب، رگرسیون لجستیک می‌تواند مرز تصمیمی غیرخطی را بیان کند.

<https://www.linkedin.com/pulse/generating-non-linear-decision-boundaries-using-logistic-d-urso/>

سوال دوم

فایل کد در پوشه ی Source Codes با عنوان P2.ipynb قرار داده شده است.

بخش a : Pre-processing

تصاویر را در گوگل درایو آپلود نمودم و سپس از گوگل کولب به آن دسترسی ایجاد کردم.

```
Mounted at /content/drive
```

تصاویر را با استفاده از دستورات glob و OpenCV باز کرده و تغییر سایز را اعمال نمودم.

```
IMG_SIZE = (128, 128)
```

بخش b : Split the data

تصاویر را به نسبت ۲۵ درصد و ۷۵ درصد به مجموعه تست و آموزش تقسیم کردم.

```
train_test_split(X, y, test_size=0.25, random_state=42)
```

بخش c : Run logistic regression

فراخوانی رگرسیون لجستیکی از کتابخانه sklearn انجام شد و با داده های آموزش، مدل ساخته شد.

بخش d : accuracy and confusion matrix

در ابتدا داده های تست را به مدل داده تا پیش بینی نماید و سپس جدول درهم آمیختگی و صحت را چاپ می نمایم.

```
Accuracy: 91.67%
Confusion Matrix:
[[ 43  16]
 [  5 188]]
```

بخش e : best probability threshold

معیار $f1_score$ را با توجه به اینکی مضربی از ویژگی های $recall$ و $precision$ است در نظر گرفته ایم و سپس داده های مختلفی را جهت بدست آوردن بهترین حالت $f1_score$ یعنی نزدیک شدن به یک را انجام می دهیم، خروجی به صورت ذیل آورده شده است.


```
Best Threshold: 0.25
Accuracy with Best Threshold: 92.06%
Confusion Matrix with Best Threshold:
[[ 42  17]
 [  3 190]]
```

بله مقدار کمی بهبود داشتیم.

بخش f

ذخیره سازی مدل

با استفاده از کتابخانه pickle مدل را ذخیره می کنیم.

 best_model.pkl	5/4/2023 1:55 AM	PKL File	385 KB
--	------------------	----------	--------

سپس مجدد مدل را با استفاده از pickle میخوانیم و قدم بعدی را یعنی خواندن مجموعه تست از گوگل کولب و تغییر سائز را آغاز می کنیم.

فایل مربوط به بخش ذخیره ی مدل در پوشه ی Source Codes با عنوان best_model.pkl قرار داده شده است.

استفاده از مدل ذخیره شده

خروجی پیش بینی تصاویر را می توانید مشاهده کنید. (در صورتی که کیفیت پایین است، فایل به عنوان ضمیمه نیز ارسال شده است.



فایل مربوط به این تصویر در پوشه ی Report با عنوان P02Result.png قرار داده شده است.

سوال سوم

فایل کد در پوشه ی Source Codes با عنوان P4.ipynb قرارداده شده است.

بخش a

خواندن داده ها

(70000, 4)

	Unnamed: 0	comment	label	label_id
0	NaN	واقعا حیف وقت که بنویسم سرویس دهنیون شده افتضاح	SAD	1.0
1	NaN	...قرار بود ۱ ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0.0
2	NaN	...قیمت این مدل اصلا با کیفیتش سازگاری نداره، فقط	SAD	1.0
3	NaN	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0.0
4	NaN	...شیرینی وانیلی فقط یک مدل بود	HAPPY	0.0

حذف ستون بدون کاربرد

	comment	label	label_id
0	واقعا حیف وقت که بنویسم سرویس دهنیون شده افتضاح	SAD	1.0
1	...قرار بود ۱ ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0.0
2	...قیمت این مدل اصلا با کیفیتش سازگاری نداره، فقط	SAD	1.0
3	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0.0
4	...شیرینی وانیلی فقط یک مدل بود	HAPPY	0.0

حذف سطرهای خالی کلی و کامنت های خالی

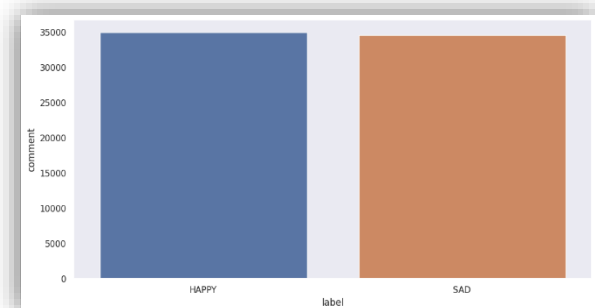
(69480, 4)

	comment	label	label_id	num_letters
0	واقعا حیف وقت که بنویسم سرویس دهنیون شده افتضاح	SAD	1.0	47
1	...قرار بود ۱ ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0.0	146
2	...قیمت این مدل اصلا با کیفیتش سازگاری نداره، فقط	SAD	1.0	89
3	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0.0	101
4	...شیرینی وانیلی فقط یک مدل بود	HAPPY	0.0	29

نصب hazm و nltk روی colab

```
!pip install hazm
```

مصور سازی داده ها بر اساس لیبل



عملیات lemmatize و normalizer

(68913, 6)

	comment	label	label_id	numberOfLetters	clean	num_clean
0	حیف وقت بنویسم سرویس دهیتون افتضاح	SAD	1	34.0	حیف وقت بنویسم سرویس دهیتون افتضاح	35.0
1	...قرار ساعته برسه نیم ساعت موقع رسید، ببین چقدر	HAPPY	0	102.0	...قرار ساعته برسه نیم ساعت موقع رسید ببین چقدر	101.0
2	...قیمت مدل کیفیتش سازگاری نداره، ظاهر فریبنده دا	SAD	1	71.0	...قیمت مدل کیفیتش سازگاری نداره ظاهر فریبنده دار	70.0
3	...عاللی اندازه کیفیت خوب، امیدوارم کیفیتتون باش	HAPPY	0	65.0	...عاللی اندازه کیفیت خوب امیدوارم کیفیتتون باشه	65.0
4	...شیرینی وانیلی مدل بود	HAPPY	0	22.0	...شیرینی وانیلی مدل بود	22.0

عملیات tokenizer روی متن

(69480, 6)

	comment	label	label_id	num_letters	clean	num_clean
0	واقعا حیف وقت که نوشت بنویس سرویس دهیتون شده افتضاح	SAD	1.0	47	...واقعا حیف وقت که نوشت بنویس سرویس دهیتون شده افتضاح	51
1	...قرار بود 1 ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0.0	146	...قرار بود 1 ساعته برسه ولی نیم ساعت زود از موقع	150
2	...قیمت این مدل اصلا با کیفیت سازگاری نداره، فقط	SAD	1.0	89	...قیمت این مدل اصلا با کیفیت سازگاری نداره، فقط	91
3	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0.0	101	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	101
4	...شیرینی نیل فقط یک مدل بود	HAPPY	0.0	29	...شیرینی نیل فقط یک مدل بود	30

حذف ستون های اضافه

(69480, 4)

	comment	label	label_id	clean
0	واقعا حیف وقت که بنویسم سرویس دهیتون شده افتضاح	SAD	1.0	...واقعا حیف وقت که نوشت بنویس سرویس دهیتون شده افتضاح
1	...قرار بود 1 ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0.0	...قرار بود 1 ساعته برسه ولی نیم ساعت زود از موقع
2	...قیمت این مدل اصلا با کیفیت سازگاری نداره، فقط	SAD	1.0	...قیمت این مدل اصلا با کیفیت سازگاری نداره، فقط
3	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0.0	...عاللی بود همه چه درست و به اندازه و کیفیت خوب
4	...شیرینی وانیلی فقط یک مدل بود	HAPPY	0.0	...شیرینی نیل فقط یک مدل بود

دانلود stopwords فارسی و حذف کلمات

(69480, 4)				
	comment	label	label_id	clean
0	واقعاً حیف وقت که بنویسم سرویس دهنیون شده اقتضاح	SAD	1.0	واقعاً حیف وقت نوشتن بنویسم سرویس دهنیون شده اقتضاح
1	...قرار بود ۱ ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0.0	...قرار بود ۱ ساعته برسه نیم ساعت زود موقع ، دید
2	...قیمت این مدل اصلاً با کیفیتش سازگاری نداره، فقط	SAD	1.0	...قیمت مدل اصلاً کیفیت سازگاری نداره ، ظاهر فرین
3	...عالی بود درست اندازه کیفیت ، امیدوارم کیفیت خوب	HAPPY	0.0	...عالی بود درست اندازه کیفیت ، امیدوارم کیفیت
4	شیرینی وانیلی فقط یک مدل بود	HAPPY	0.0	شیرینی نل مدل بود

لیبل انکودینگ

انجام شد.

تفکیک داده تست و آموزش

با نسبت ۲۵ درصد تست و ۷۵ درصد آموزش انجام شد.

استفاده از CountVectorizer و TFIDFVectorizer

از CountVectorizer برای بدست آوردن ۱۵۰ کلمه پرتکرار به صورت تک کلمه ای استفاده کردیم و همچنین از

TFIDFVectorizer نیز هم.

سپس یک وکتور نهایی ساختیم.

```
150
'هات'، 'های'، 'هزینه'، 'همراه'، 'پایین'، 'پنیر'، 'پول'، 'پیتزا'، 'پیتزای'، 'پیشه‌ها'، 'یک'، 'پیگیری'، 'چرب'، 'کباب'، 'کنسرو'، 'کهنه'، 'کیفیت'، 'کیفیتش'، 'کیک'، 'گرم'، 'گرچه'، 'گوشت'، 'بخ'، 'یکم'،
<
```

حذف محتویات اضافه در متن ها

(65189, 5)				
	comment	label	label_id	clean
0	حیف وقت بنویسم سرویس دهنیون اقتضاح	SAD	1	حیف وقت بنویسم سرویس دهنیون اقتضاح
1	...قرار ساعته برسه نیم ساعت موقع رسید، بین چقدر	HAPPY	0	...قرار ساعته برسه نیم ساعت موقع رسید بین چقدر
2	...قیمت مدل کیفیتش سازگاری نداره، ظاهر فریبده دا	SAD	1	...قیمت مدل کیفیتش سازگاری نداره ظاهر فریبده دار
3	...عالی اندازه کیفیت خوب، امیدوارم کیفیتتون باش	HAPPY	0	...عالی اندازه کیفیت خوب امیدوارم کیفیتتون باشه
4	شیرینی وانیلی مدل بود	HAPPY	0	شیرینی وانیلی مدل بود

بخش b

توضیحات

خب میایم و پس از پیش پردازش داده ها اقدام می کنیم به اینکه کلمات پرتکرار رو به صورت مجموعه داده ها رو بیرون میاریم و لیبل میزنیم و بعدش لیبل رو در هر مجموعه اضافه میکنیم و با این کار یک بردار می سازیم. برای فیت کردن میایم بردار ها رو بر اساس فرمول بندی اساسی NaiveBayes که در لینک آورده شده بود استفاده می کنیم.

برای تست نیز هر جمله ی ورودی رو اقدام به وکتور سازی ازش می کنیم و در نهایت وکتور رو بررسی نموده و خروجی را مشاهده می کنیم.

پیاده سازی تابع fit

```
def fit(x, y, labels):
    n_label_items = {}
    log_label_priors = {}
    n = len(x)
    grouped_data = group_by_label(x, y, labels)
    for l, data in grouped_data.items():
        n_label_items[l] = len(data)
        log_label_priors[l] = math.log(n_label_items[l] / n)
    return n_label_items, log_label_priors
```

پیاده سازی تابع predict

```
def predict(n_label_items, vocab, word_counts, log_label_priors, labels, x):
    result = []
    for text in x:
        label_scores = {l: log_label_priors[l] for l in labels}
        words = set(w_tokenizer.tokenize(text))
        for word in words:
            if word not in vocab: continue
            for l in labels:
                log_w_given_l = laplace_smoothing(n_label_items, vocab, word_counts, word, l)
                label_scores[l] += log_w_given_l
        result.append(max(label_scores, key=label_scores.get))
    return result
```

استفاده از GNB

داده ها را به وکتور تبدیل کردیم طی فرایندی و از خود GNB تعریف شده در sklearn استفاده نمودیم.

(65189, 7)							
	comment	label	label_id	clean	newText	Vector	VectorLen
0	حیف وقت بنویسم سرویس دهیتون اقتضاح	SAD	1	حیف وقت بنویسم سرویس دهیتون اقتضاح	اقتضاح	{7}	1.0
1	...قرار ساعته برسه نیم ساعت موقع رسید، ببین چقدر	HAPPY	0	...قرار ساعته برسه نیم ساعت موقع رسید ببین چقدر	نیم ساعت موقع رسید مزه	{66, 101, 107, 125, 61}	5.0
2	...قیمت مدل کیفیت سازگاری نداره، ظاهر فریبده دا	SAD	1	...قیمت مدل کیفیت سازگاری نداره ظاهر فریبده دار	قیمت کیفیت نداره قارچ	{144, 97, 96, 117}	4.0
3	...عاللی اندازه کیفیت خوب، امیدوارم کیفیتتون باش	HAPPY	0	...عاللی اندازه کیفیت خوب امیدوارم کیفیتتون باشه	اندازه کیفیت باشه مشتری	{10, 13, 102, 143}	4.0
4	...شیرینی واتیلی مدل بود	HAPPY	0	...شیرینی واتیلی مدل بود	شیرینی بود	{83, 23}	2.0

بخش C

استفاده از NB داده شده

```

Accuracy of prediction on test set : 0.5463860596392195
precision_score of prediction on test set : 0.5291881443298969
recall_score of prediction on test set : 0.9896373056994818
f1_score of prediction on test set : 0.6896175322221756
classification_report of prediction on test set :
      precision    recall  f1-score   support

      0       0.89       0.09       0.16       7999
      1       0.53       0.99       0.69       8299

 accuracy          0.55          16298
 macro avg       0.71          16298
 weighted avg    0.71          16298

```

SKlearn در GNB

```

Number of mislabeled points out of a total 19557 points : 8454
Accuracy of prediction on test set : 0.5677251112133763
precision_score of prediction on test set : 0.6124685325040723
recall_score of prediction on test set : 0.41471974330692873
f1_score of prediction on test set : 0.4945593686476146
classification_report of prediction on test set :
      precision    recall  f1-score   support

      0       0.54       0.73       0.62       9584
      1       0.61       0.41       0.49       9973

 accuracy          0.57          19557
 macro avg       0.58          19557
 weighted avg    0.58          19557

```

سوال چهارم

فایل کد در پوشه ی Source Codes با عنوان P4.ipynb قرارداده شده است.

بخش a

دیتاست لود شد و به دو بخش X و y یا همان تارگت تقسیم شد. سپس با نسبت ۰,۷ به ۰,۳ به آموزش و تست تخصیص داده شد.

بخش b

داده ها را با نوشتن کد نویزی می کنیم، یک نمونه تصویر پس از نویزی شدن :



بخش c

هدف این است ما بتوانیم نویز را کاهش دهیم، طبیعتاً می‌خواهیم نزدیک ترین تصاویر ممکن را بیابیم و بر اساس آن نزدیک ترین ها که فاقد نویز هستن، تصویر نویزی را رفع نویز کنیم.

بخش d

انتخاب مدل

از KNN استفاده کردیم.

اسکیل کردن

تصاویر را با MinMax اسکیل کردیم تا همیشه صرفاً از Standard استفاده نکرده باشیم.

ساخت مدل و ارزیابی اولیه

مدل را ساختیم و سپس تصاویر بدون نویز را برای آموزش به طبقه بند دادیم، سپس ارزیابی ذیل را بدست آوردیم.

	precision	recall	f1-score	support
0	0.92	0.99	0.95	2071
1	0.98	0.98	0.98	2363
2	0.97	0.95	0.96	2097
3	0.94	0.92	0.93	2142
4	0.98	0.89	0.94	2047
5	0.96	0.88	0.92	1894
6	0.97	0.97	0.97	2063
7	0.97	0.94	0.95	2188
8	0.86	0.94	0.90	2048
9	0.88	0.93	0.90	2087
accuracy			0.94	21000
macro avg	0.94	0.94	0.94	21000
weighted avg	0.94	0.94	0.94	21000

پیدا کردن همسایه

در ادامه تصاویر نویزی را برای پیش بینی به مدل دادیم و همسایه های هر کدام از تصاویر نویزی را یافتیم.

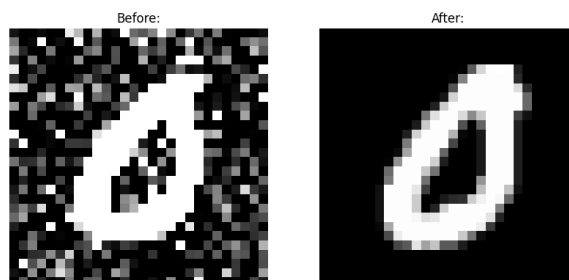
در نهایت از همسایه ها برای بازسازی استفاده کردیم.

```
y_kneighbors = knn.kneighbors(X_test_noisy, return_distance=False)
```

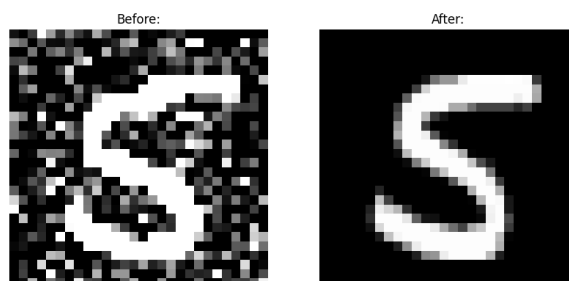
بخش e

چند نمونه از خروجی کارها در ادامه قرار داده شده است:

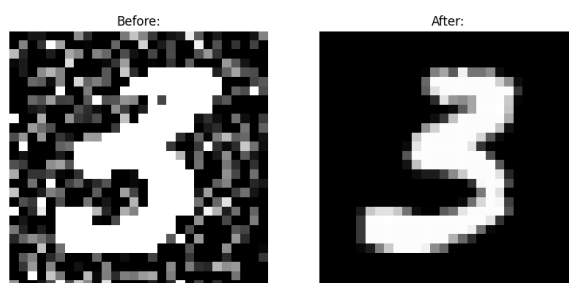
نمونه اول



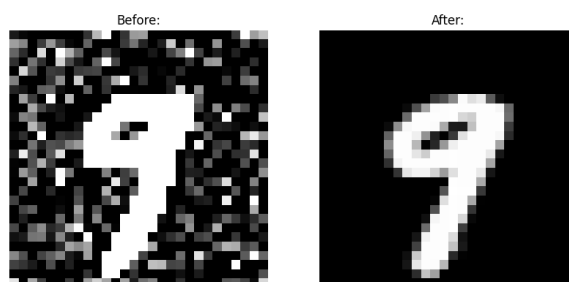
نمونه دوم



نمونه سوم



نمونه چهارم



نمونه پنجم

اشتباه (۳ را ۸ کرده است).

