گزارش پیاده‌سازی الگوریتم‌های
Least Square (closed form)
و
Batch Gradient Descent

محمد شعاعی

۳۰ آبان ۱۴۰۰

بخش اول
توابع و کلاس‌ها

پیاده سازی الگوریتم‌ها با استفاده از زبان برنامه‌نویسی پایتون[۱] انجام شده است. فایل‌های مربوط به این گزارش عبارتند از:

۱. preprocess.py: این فایل شامل ۲ تابع برای نرمال‌سازی داده و ۱ تابع برای محاسبه خطا است.

(آ) `zero_mean_normalize(df: pd.DataFrame, columns: list)` : این تابع ستون‌های مشخص شده را نرمال‌سازی می‌کند. اجرای این تابع داده‌ی اولیه را تغییر می‌دهد.

(ب) `min_max_normalize(df: pd.DataFrame, columns: list)` : این تابع با استفاده از روش کمینه ـ بیشینه داده‌ها را نرمال می‌کند. این تابع داده اولیه را تغییر می‌دهد.

(ج) `mean_squared_error(y_true, y_pred)` : این تابع میانگین مربع خطا را به صورت یک عدد اعشاری باز می‌گرداند.

۲. regressors.py: این فایل شامل ۳ کلاس مربوط به پیاده سازی الگوریتم های مختلف رگرسیون خطی است.

(آ) `class LeastSquareRegressor` : این کلاس شامل پیاده سازی فرم بسته[۲] است.

(ب) `class BGDRegressor` : این کلاس شامل پیاده سازی الگوریتم گرادیان کاهشی دسته‌ای[۳] است.

---

[۱] python
[۲] closed form
[۳] Batch Gradient Descent

(ج) `class SGDRegressor` : این کلاس شامل پیاده سازی الگوریتم گرادیان کاهشی تصادفی[۴] است.

۳. main.py: این فایل که به صورت رابط کاربری متنی پیاده سازی شده است از کلاس‌ها و توابعی که پیش از این معرفی شدند استفاده کرده و اطلاعات مورد نیاز مربوط به داده‌های ورودی را در خروجی استاندارد چاپ می‌کند. همچنین نمودارهای مربوط به هر الگوریتم را به صورت فایل ذخیره می‌کند. نحوه‌ی اجرای این فایل در کد ۱ نمایش داده شده است

```
$ python main.py --algo=all --normalize=zero-mean
↪  --train=Data-Train.csv --test=Data-Test.csv
```

کد ۱: اجرای فایل main.py

در ادامه دو کلاس فرم بسته و گرادیان کاهشی دسته‌ای توضیح داده خواهند شد. همچنین نمودار های مربوط به عملکرد هر یک نمایش داده می‌شود.

بخش دوم
Least Square (closed form)

این الگوریتم توسط کلاس `LeastSquareRegressor` پیاده‌سازی شده است. این کلاس شامل متدهای زیر است:

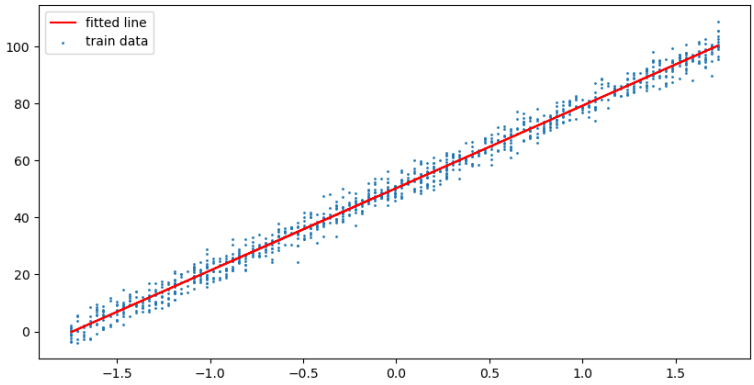
۱. `fit(self, X, y)` : با اجرای این متد روند یادگیری مدل آغاز شده و ضرایب مربوط به θ در متغیر `theta_` ذخیره می‌شوند.

۲. `predict(self, X)` : این متد برای استفاده از مدل و پیش‌بینی برچسب کاربرد دارد.

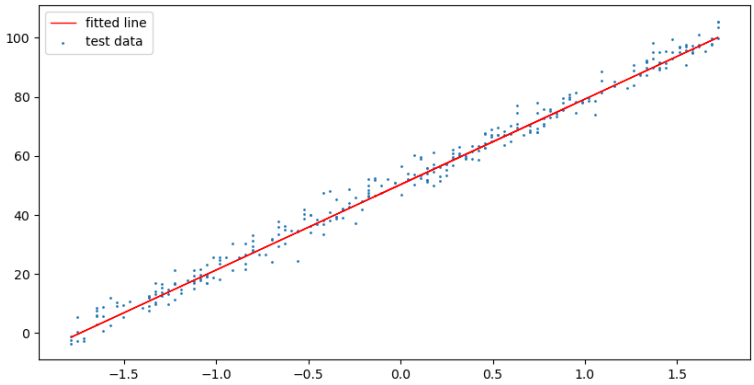
```
$ python main.py --algo=lstsq --normalize=zero-mean
↪  --train=Data-Train.csv --test=Data-Test.csv

theta: [50.29964125 28.93785624]
coefs: [28.93785624]
y-intercept: 50.299641248803006
MSE (train): 8.328012371573903
MSE (test): 9.98467535752803
```

کد ۲: اجرای الگوریتم فرم بسته

[۴] Descent Gradient Stochastic



شکل ۱: نمودار مربوط به داده train



شکل ۲: نمودار مربوط به داده test

بخش سوم
Batch Gradient Descent

این الگوریتم با محاسبه‌ی گرادیان مربوط به تمام نمونه‌های موجود و سپس محاسبه‌ی خطا و پس از آن به روزرسانی مقادیر مربوط به θ اقدام به یافتن خط با کمترین خطا می‌کند. این الگوریتم توسط کلاس `BGDRegressor` پیاده‌سازی شده است. این کلاس شامل متدهای زیر است:

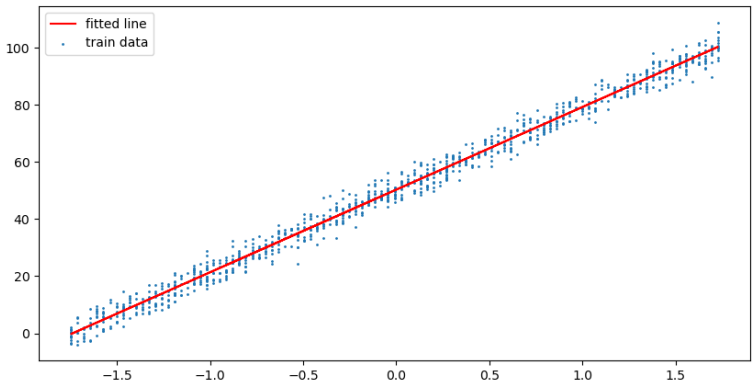
۱. `fit(self, X, y, eta=0.0001, n_iters=1e2)` : اجرای این متد روند یادگیری مدل را آغاز می‌کند. این متد فرآیند محاسبه‌ی خطا و بروزرسانی θ را به تعداد `n_iter` مرتبه انجام می‌دهد و در هر مرحله مقدار خطا (هزینه) را در متغیر `costs_` ذخیره می‌کند.

۲. `predict(self, X)` : این متد برای استفاده از مدل و پیش‌بینی برچسب .کاربرد دارد

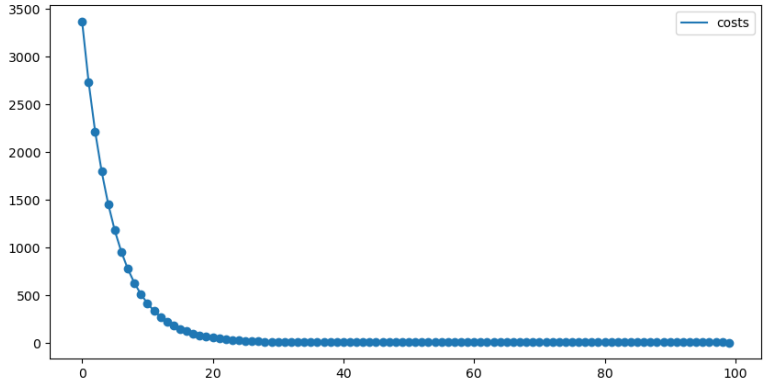
۳. `cost(self, X, y)` : این متد مقدار هزینه با توجه به θ فعلی را برمی‌گرداند.

```
$ python main.py --algo=bgd --normalize=zero-mean
↪  --train=Data-Train.csv --test=Data-Test.csv

theta: [50.29830521996789, 28.93707902574484]
coefs: [28.93707903]
y-intercept: 50.29830521996789
MSE (train): 8.328014760009802
MSE (test): 9.987058395713497
```

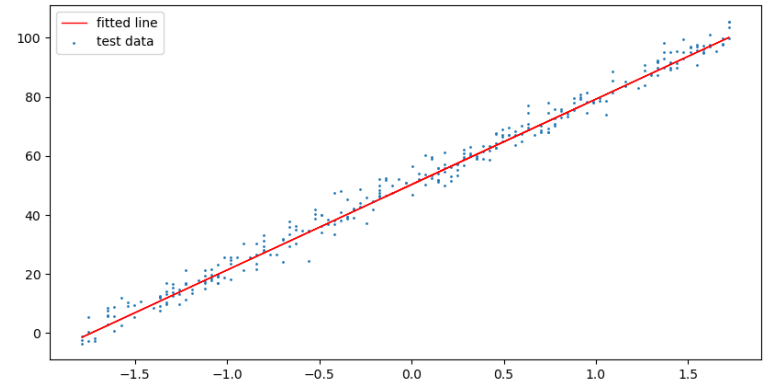
کد ۳: اجرای الگوریتم گرادیان کاهشی دسته‌ای



شکل ۳: نمودار مربوط به داده train



شکل ۴: نمودار مربوط به هزینه



شکل ۵: نمودار مربوط به داده test