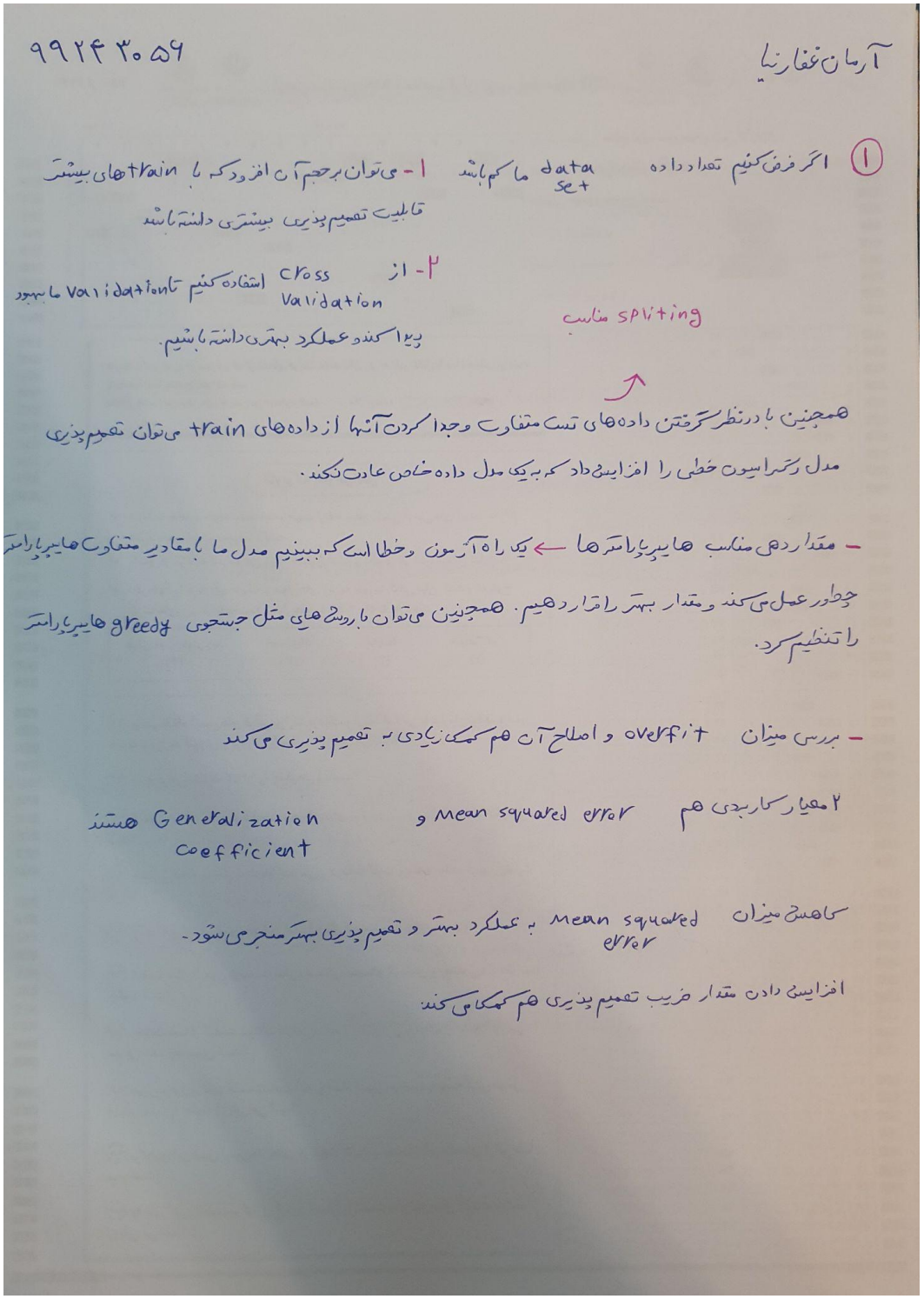
گزارش تمرین دوم آرمان غفارنیا 99243056

تحلیلی :

1)



(2 و 3)

(۲) ۲ روش وجود دارد که بتوان روش دسته‌بندی دودویی را به حالت چندگانه تصمیم داد.

one-Vs-one ← در این روش از بین دسته‌های موجود انتخاب های ۲تایی ممکن را انجام می‌دهیم و برای هرکدام از آنها یک مدل دودویی آموزش داده می‌شود و پیش بینی آن به عنوان یک رأی در نظر گرفته می‌شود. مثلاً

دسته ۱ و ۴ ← رأی x
دسته ۲ و ۴ ← رأی y
⋮

در نهایت بیشترین رأی پذیرفته و انتخاب می‌شود.

one-Vs-rest ← در این روش برای هر دسته یک مدل دودویی آموزش می‌بیند به این معنا که یکبار دسته موجود جای می‌گیرند سایرین در یک دسته دیگر

مدلی که بیشترین دقت را داشته باشد به عنوان پیش‌بینی نهایی انتخاب می‌شود.

OVO ← مناسب برای وقتی که دیتای کمی برای آموزش داریم.

OVR ← مناسب برای وقتی که تعداد دسته ها زیاد است.

(۳) همانطور که در شکل مشخص است نمی‌توان یک روند خطی پیدا کرد که براساس داده‌های ورودی و مقادیر y پیش بینی مناسبی انجام دهد. (پیش بینی به کمک خط ها قابل انجام نیست و اشکال غیرخطی‌اند)

اگر امتیاز زیادی به مدل خطی کنیم. ممکن است در اثر آموزش زیاد مدل دچار overfit شود و برای رسیدن به دقت خوب فقط به همین مدل عادت کند و در داده‌های دیگر عملکرد بدی داشته باشد.

درغیر اینصورت هم دقت مناسبی نخواهیم داشت

(4

(۴) در مواجهه با داده‌های نامتوازن (تعداد نمونه‌های مختلف بین دسته‌ها متفاوت است) می‌توان از چند روش استفاده کرد تا مدل ما عملکرد بهتری داشته باشد

Class weighting ← در این روش به دسته‌هایی که نمونه‌های کمتری دارند وزن بالاتری داده می‌شود تا تأثیر لازم را در مدل ما داشته باشند.

- همچنین می‌توانیم از الگوریتم‌هایی مثل Random Forest استفاده کنیم که در مواجهه با داده‌های نامتوازن خوب عمل می‌کنند و تأثیر کلاس‌های کم‌حجم مناسب باشد.

- برای ارزیابی داده‌های نامتوازن از Roc curve استفاده کرد

- همچنین می‌توان داده‌ها را به حالت متوازن درآورد یعنی یا به نمونه‌های موجود در دسته‌هایی با نمونه کم نمونه اضافه شود یا از دسته‌هایی که نمونه بیشتری دارند نمونه کم کرد.

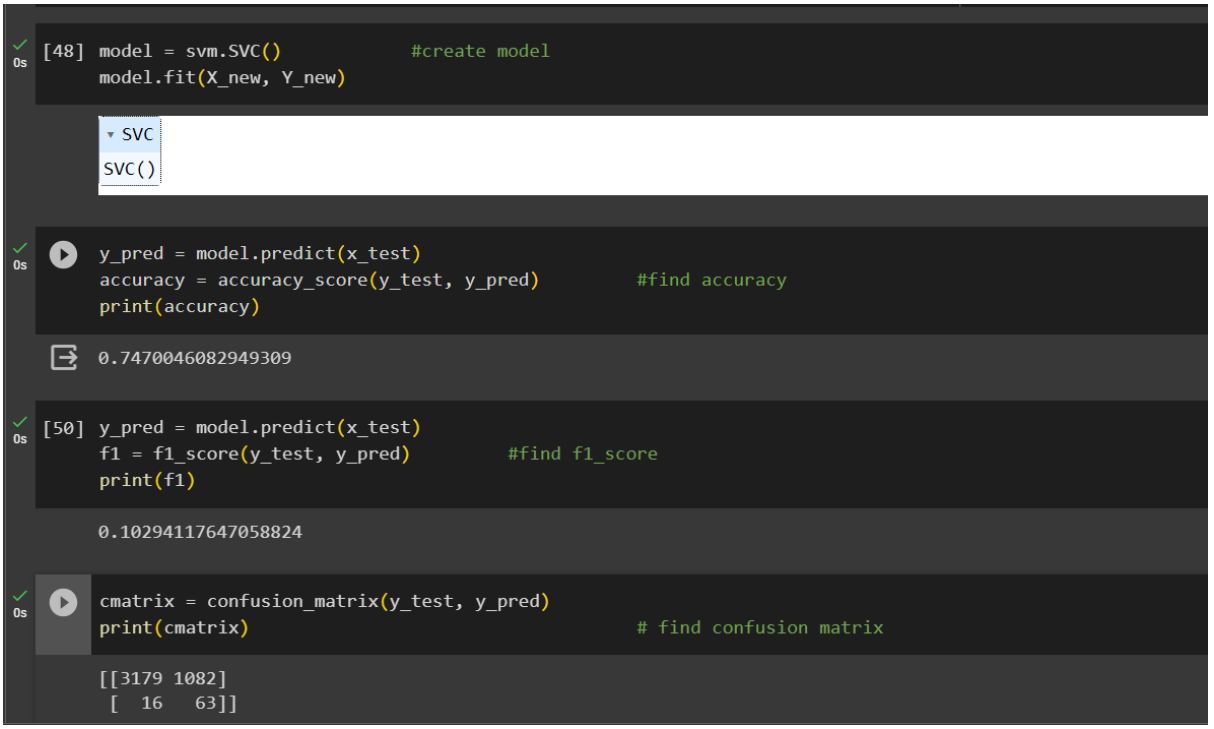
(Duplication) ← با استفاده از این روش می‌توان نمونه‌ها را بیشتر کرد.

کدی :

1)

الف) در کد انجام شد.

ب و پ)



آموزش داده شد.
مقدار accuracy برابر 74 درصد و مقدار f1_score برابر با 10 درصد شد.
همچنین مقادیر کانفیوژن ماتریس هم آورده شد.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

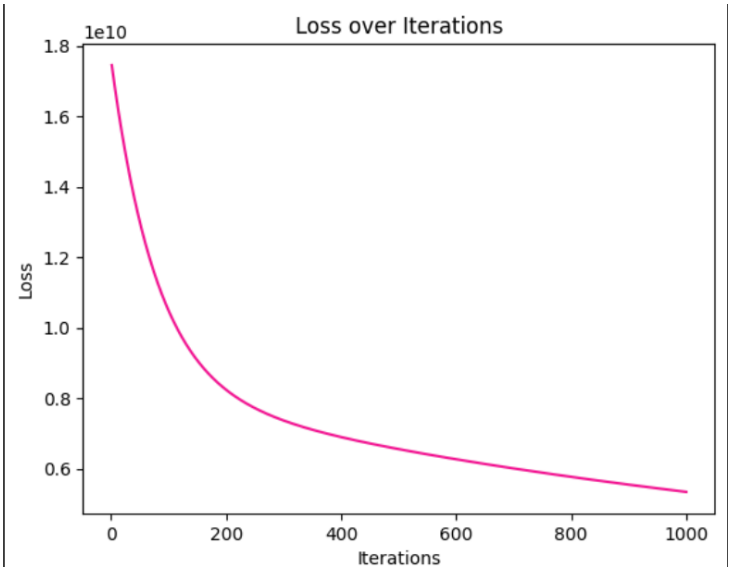
$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

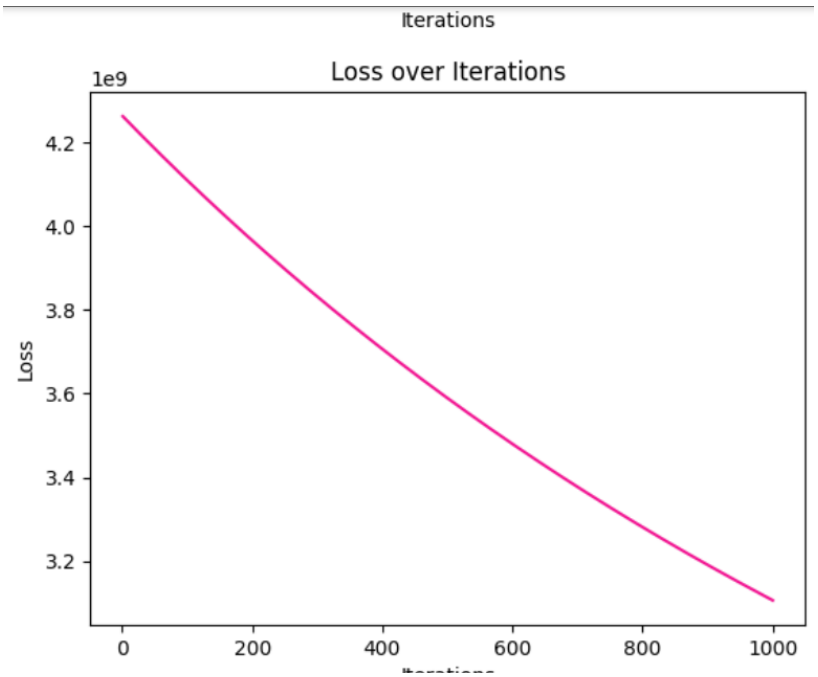
2)

الف) در کد انجام شد.

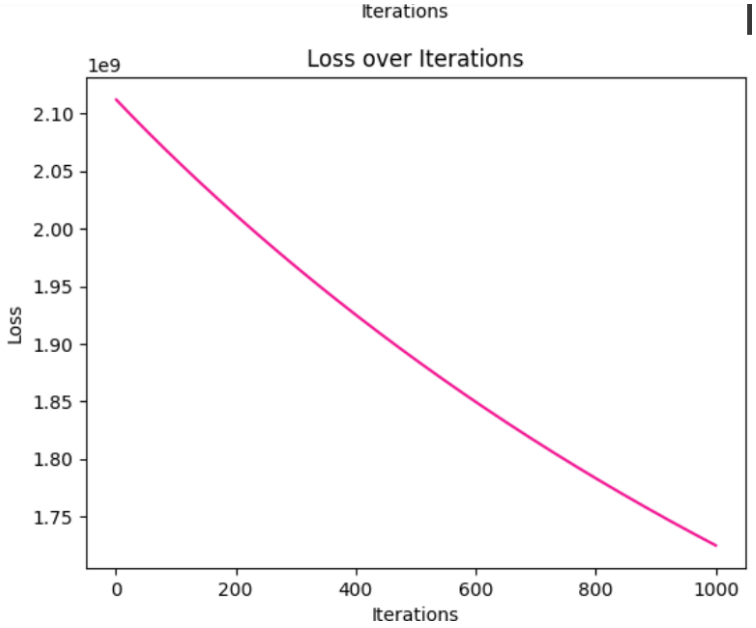
ب)
نمودار loss را برای درصد اسپلیت های متفاوت رسم میکنیم



10 درصد داده تست



30 درصد داده تست



60 درصد داده تست

مشاهده میشود که در 10 درصد داده تست هم نرخ کاهش بیشتری دارد هم در نهایت به مقدار کمتری رسیده است (به نسبت cost اولیه) پس عملکرد بهتری دارد.

پ)

پیاده سازی در کد انجام شد . دقت را با r-square error گزارش میکنیم

```
predict = np.dot(x1_test, W) + B
r_square = r2_score(y1_test, predict)    # find r-squre error
print(r_square)
```

```
0.6145017246529894
```

$$R^2 = 1 - \frac{\text{Sum of Squared Residuals}}{\text{Total Sum of Squares}}$$

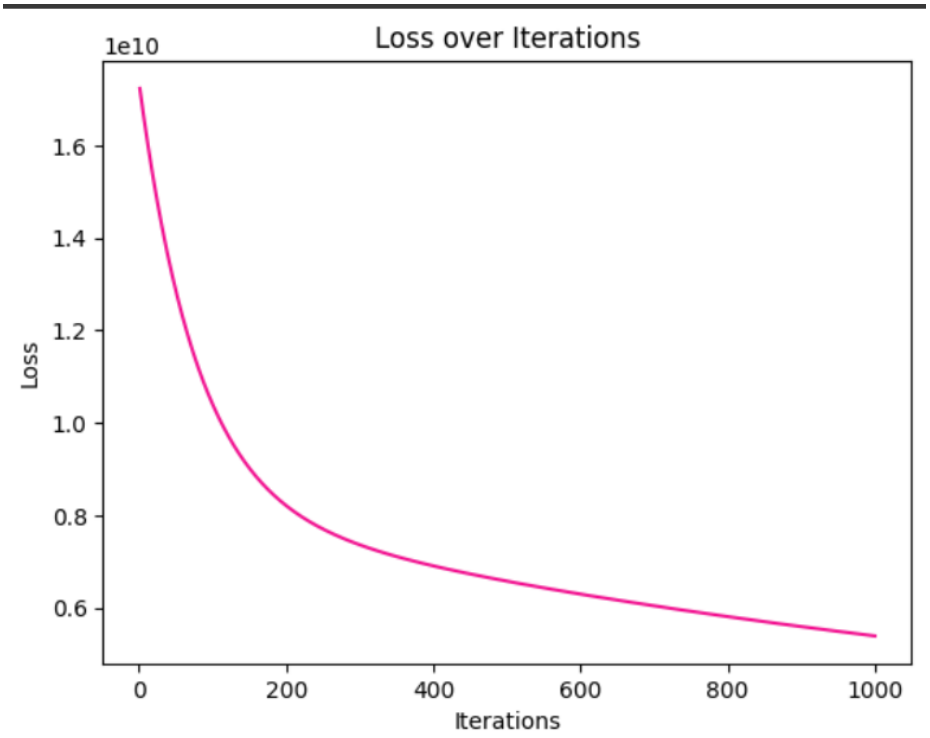
where:

- Sum of Squared Residuals (SSR) is the sum of the squared differences between the actual values and the predicted values.
- Total Sum of Squares (SST) is the sum of the squared differences between the actual values and the mean of the actual values.

ث)

```
data_set['region_0_2'] = data_set["region_0"]**2
data_set['region_1_2'] = data_set["region_1"]**2
data_set['region_2_2'] = data_set["region_2"]**2
data_set['region_3_2'] = data_set["region_3"]**2

data_set['smoker_0_2'] = data_set["smoker_0"]**2
data_set['smoker_1_2'] = data_set["smoker_1"]**2          # add polonomials

data_set['sex_0_2'] = data_set["sex_0"]**2
data_set['sex_1_2'] = data_set["sex_1"]**2

data_set['bmi2'] = data_set["bmi"]**2

data_set['children2'] = data_set["children"]**2
```

فیچر های دیتا را به توان 2 رسانده و به دیتا اضافه میکنیم.



مشاهده میشود مجدد loss فانکشن نزولی است و نرخ کاهش خوبی دارد و به مقدار خوبی در نهایت رسیده است (به نسبت cost اولیه)

```
predict = np.dot(x1_test, W) + B
r_square = r2_score(y1_test, predict)     # find r-squre error
print(r_square)
```

```
0.6518596626251604
```

همچنین دقت مدل در حالت چندجمله ای بهتر شد.