به نام خدا

نام : پریا جبرئیلی شماره دانشجویی:993212018

در بخش اول با توجه به شماره دانشجویی 10 ستون به صورت رندوم از کل ستونهای دیتاست انتخاب شد. با کمک این ستونها یک دیتاست جدید ایجاد کردم.

```
['k', 's', 'd', 't', 'o', 'h', 'f', 'p', 'c', 'e']
```

['Gender', 'Customer Type', 'Age', 'Type of Travel', 'Flight Distance', 'Ease of Online booking', 'Seat comfort', 'Inflight entertainment', 'Baggage handling', 'Checkin service', 'satisfaction']

	Gender	Customer Type	Age	Type of Travel	Flight Distance	Ease of Online booking	Seat comfort	Inflight entertainment	Baggage handling	Checkin service	satisfaction
62088	Male	disloyal Customer	30	Business travel	693	0	2	2	5	5	satisfied
43084	Male	disloyal Customer	9	Business travel	821	3	3	5	3	5	satisfied
50947	Female	Loyal Customer	9	Business travel	1562	3	1	3	4	2	neutral or dissatisfied
96181	Male	Loyal Customer	12	Personal Travel	473	5	4	4	4	3	satisfied
71525	Female	disloyal Customer	22	Business travel	214	3	5	5	4	3	neutral or dissatisfied
70234	Male	Loyal Customer	59	Business travel	1061	0	5	5	4	3	neutral or dissatisfied

در بخش بعدی چک کردم اگر missing value ای وجود دارد یا خیر.

این مقادیر از دست رفته احتمالاً به این دلیل از دست رفته اند که ثبت نشده اند، نه به دلیل اینکه وجود ندارند.

اگر missing value وجود داشته باشد میتواند drop شود یا اینکه می توان مقادیرش با مقدار منطقی دیگری پر کرد. این مقدار رو میتونیم خودمون بدیم یا مقدار سطر بعدی یا قبلی همون ستون رو بهش نسبت بدهیم.

که هیچ missing value ای نداشتم.

```
Gender                          0
Customer Type                   0
Age                             0
Type of Travel                  0
Flight Distance                 0
Ease of Online booking          0
Seat comfort                    0
Inflight entertainment          0
Baggage handling                0
Checkin service                 0
satisfaction                    0
dtype: int64
```

در بخش بعدی سعی کردم دیتا هایی که تکراری هستند را مدیریت کنم.

```
flight_satisfaction = flight_satisfaction.drop_duplicates()
```
✓ 0.1s

در بخش بعدی به شناسایی دیتاهای پرت یا outliers با visualization پرداختم.

Outlier چیست؟

Outlier به نقاط داده‌ای گفته می‌شود که به طور قابل توجهی متفاوت از سایر داده‌ها هستند. در واقع، اوتلایرها نقاطی در مجموعه داده هستند که از الگوی کلی داده‌ها خارج بوده و ممکن است نشان‌دهنده تغییرات غیرعادی یا خطاهای اندازه‌گیری باشند.

وجود اوتلایرها می‌تواند بر تحلیل‌های آماری و مدل‌های پیش‌بینی تأثیر بگذارد، زیرا ممکن است باعث تحریف نتایج و برآوردهای نادرست شوند. به عنوان مثال، یک اوتلایر می‌تواند میانگین مجموعه داده‌ها را به شکل قابل توجهی تغییر دهد.

شناسایی و رسیدگی به اوتلایرها یک قسمت مهم در پردازش و تحلیل داده‌ها است. رویکردهای مختلفی برای شناسایی اوتلایرها وجود دارد، از جمله استفاده از نمودارهای آماری مانند جعبه‌ای (Boxplot) یا تحلیل فاصله‌های بین‌کوارتیلی (IQR).

یکی از راحت ترین روشها برای اینکه تشخیص بدهیم دیتا ما دارای outlier هست یا خیر plot کردن دیتا موردنظر هست.

با کمک روشهایی این outlier ها را تشخیص میدهیم و انها را از دیتا حذف میکنیم و با دیتاست جدید کار میکنیم.

- یکی از این روشها z-score هست.

z-score چیست؟

z-score، یک معیار آماری است که نشان می‌دهد هر داده‌ای نسبت به میانگین مجموعه داده‌ها چقدر دور یا نزدیک است. به بیان دیگر، z-score مشخص می‌کند که یک نقطه داده خاص چند انحراف معیار بالاتر یا پایین‌تر از میانگین قرار دارد.

یکی از کاربردهای رایج z-score در شناسایی اوتلایرها (نقاط داده‌ای که به شدت از سایر داده‌ها متفاوت هستند) است.

بنابراین، z-score ابزاری قدرتمند در تحلیل آماری است که به ما اجازه می‌دهد داده‌ها را به شکل استاندارد بررسی کرده و نقاط غیرعادی را شناسایی کنیم.

- یکی دیگر از روشها IQR هست.

IQR چیست؟

IQR اختلاف بین کوارتیل سوم (Q3) و کوارتیل اول (Q1) در یک مجموعه داده است. به بیان ساده‌تر، IQR نشان‌دهنده محدوده‌ای از داده‌ها است که نیمی از کل داده‌ها در آن قرار دارند.

IQR محاسبه می‌شود به صورت: $IQR = Q3 - Q1$

برای شناسایی اوتلایرها با استفاده از IQR، معمولاً از 1.5 برابر IQR فراتر از کوارتیل‌های اول و سوم استفاده می‌شود:

$Upper\ Bound = Q3 + 1.5 * IQR$

$Lower\ Bound = Q1 - 1.5 * IQR$

داده‌هایی که فراتر از این حدود قرار دارند، به عنوان اوتلایر در نظر گرفته می‌شوند.

- یکی دیگر از روشها percentile method هست .

Percentile method چیست؟

Percentile Methodیکی از روش‌های آماری است که برای شناسایی موقعیت یک نقطه داده در مقایسه با کل مجموعه داده‌ها استفاده می‌شود. Percentile Method مقادیری هستند که داده‌های یک مجموعه را به صد بخش مساوی تقسیم می‌کنند. به بیان دیگر، یک Percentile Method نشان‌دهنده مقداری است که یک درصد خاص از داده‌ها کمتر یا برابر با آن مقدار هستند.

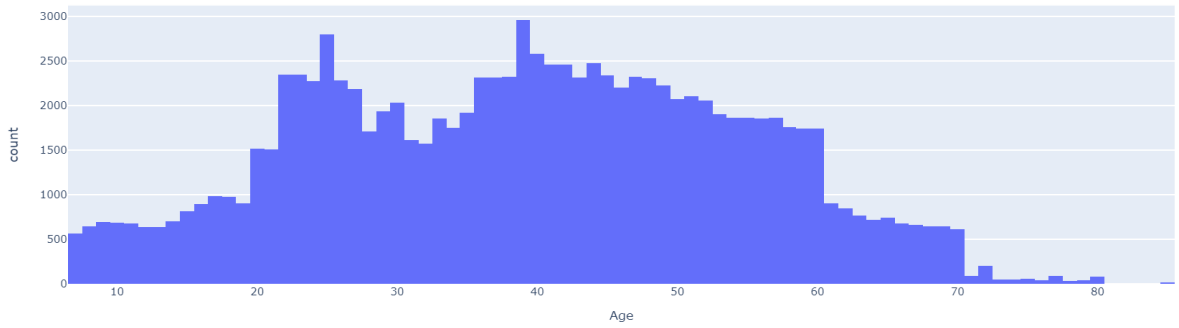
برای ستونهای زیر ابتدا داده رو plot کردم و سپس outlier ها را تشخیص دادم واونها رو حذف کردم.

برای هر کدام از ستونها هر 3 روش را برای تشخیص و حذف outlier استفاده کردم.

= Age

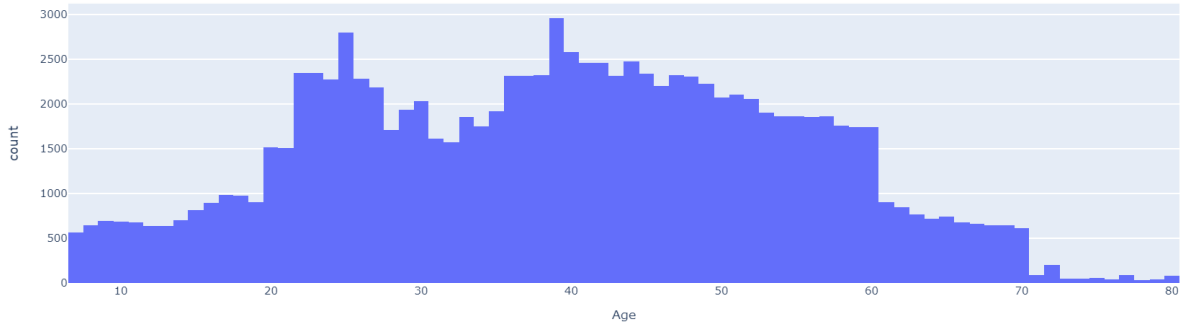
با توجه به نمودار متوجه میشویم یه سری بالاتر از 80 هستن و از داده های دیگر پرت هستند.

در این بخش از روش z-score استفاده کردم چون این توزیع تقریباً نرمال به نظر می رسد و دراین بخش z-score برای نرمال سازی خوبتر به نظر میاد.



بعد حذف outlier ها :

داده های پرت حذف شده است.



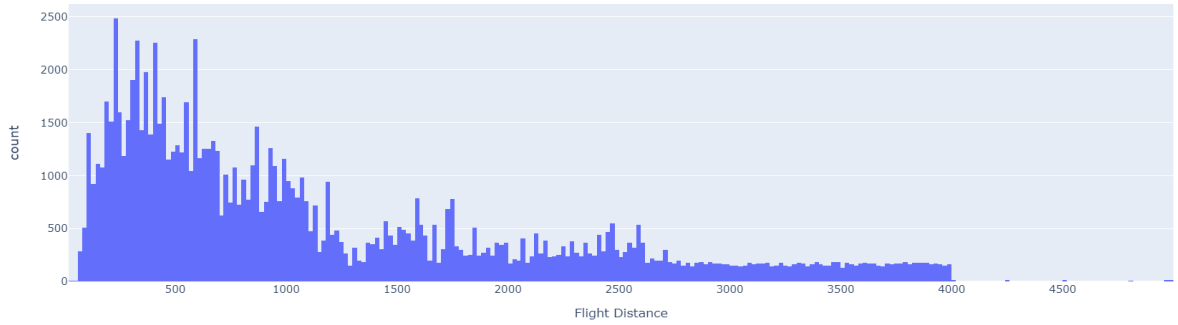
البته تعداد قبل حذف outlier و بعد حذف outlier و تعداد outlier برای Age را در این بخش پرینت کردم.

```
before removing outliers : 103847
after removing outliers : 103830
outliers : 17
```

= Flight Distance

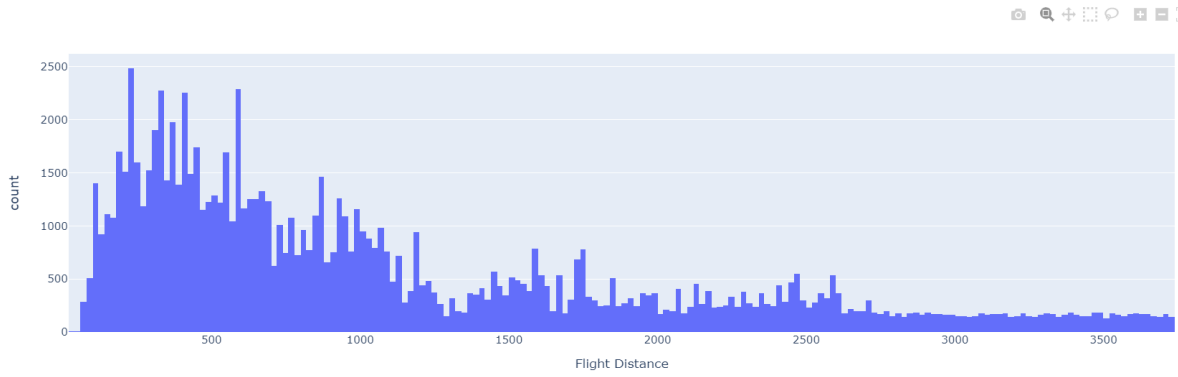
با توجه به نمودار به وضوح میتوان داده های پرت را دید که در فاصله بالاتر از 4000 هستند.

در این بخش از روش IQR استفاده کردم چون این توزیع skewed distribution (توزیع اریب)به نظر می رسد و دراین بخش IQR برای نرمال سازی خوبتر به نظر میاد.



بعد حذف outlier ها :

داده های پرت حذف شده است.



البته تعداد قبل حذف outlier و بعد حذف outlier و تعداد outlier برای Flight Distance را در این بخش پرینت کردم.

```
before removing outliers : 103847
after removing outliers : 101557
outliers : 2290
```

در بخش بعدی تبدیل نوع داده ها هست.

چرا تبدیل نوع داده‌ها انجام می‌شود و توضیح آن :

Data Type Conversion فرایندی است که در آن نوع داده‌ها در یک مجموعه داده تغییر می‌کند. این تغییر معمولاً برای اطمینان از سازگاری داده‌ها با نیازهای تحلیلی یا الزامات نرم‌افزاری خاص انجام می‌شود.دلایل و اهمیت این است که :

تغییر نوع داده‌ها می‌تواند به بهبود کارایی پردازش داده‌ها کمک کند. به عنوان مثال، تبدیل داده‌های متنی به دسته‌بندی‌های عددی می‌تواند در سرعت پردازش و تحلیل داده‌ها مؤثر باشد.

تبدیل نوع داده‌ها به استانداردسازی داده‌ها در میان مجموعه‌های مختلف کمک می‌کند.

تغییر نوع داده‌ها می‌تواند به سهولت بیشتر در تفسیر و تجزیه و تحلیل آن‌ها کمک کند، به ویژه در مواردی که نوع داده‌ها به صورت اولیه مناسب تحلیل نیست.

ابتدا نوع داده هایی که داشتم رو در خروجی چاپ کردم:

```
Gender                    object
Customer Type             object
Age                        int64
Type of Travel            object
Flight Distance            int64
Ease of Online booking     int64
Seat comfort               int64
Inflight entertainment     int64
Baggage handling           int64
Checkin service            int64
satisfaction              object
dtype: object
```

ستونهایی که نوع داده شون object بود رو به کمک numeric encoding به int تبدیل کردم.

دلیل اینکه از numeric encoding استفاده کردم این بود که شبکه عصبی ای که در بخش اخر استفاده شده نیاز به داده های عددی دارد و نمیتواند مستقیما با داده های متنی کار کند.

پس ستونهای gender و customer_Type و Type_of_Travel و satisfaction را به کمک numeric encoding به نوع داده int تبدیل میکنیم.

```
# Define a mapping for the encoding
gender_mapping = {'Male': 0, 'Female': 1}
Customer_Type_mapping={'Loyal Customer':0, 'disloyal Customer':1}
Type_of_Travel_mapping ={'Personal Travel':0, 'Business travel':1}
satisfaction_mapping={'satisfied':1, 'neutral or dissatisfied':0}
```

خروجی نوع داده به این صورت میشود:

```
Age                        int64
Flight Distance            int64
Ease of Online booking     int64
Seat comfort               int64
Inflight entertainment     int64
Baggage handling           int64
Checkin service            int64
Gender                     int64
Customer Type              int64
Type of Travel             int64
satisfaction               int64
```

در بخش بعدی Min-Max Scaling را انجام دادم. این روش برای مقیاس‌بندی داده‌ها به گونه‌ای استفاده می‌شود که مقادیر هر ویژگی در بازه 0 تا 1 قرار گیرند.

تعیین مقادیر کمینه و بیشینه برای Flight Distance=

- min_val = final_dataset['Flight Distance'].min: این خط کمینه (کوچک‌ترین مقدار) ستون 'Flight Distance' در دیتاست `final_dataset` را محاسبه و در `min_val` ذخیره می‌کند.

- max_val = final_dataset['Flight Distance'].max: این خط بیشینه (بزرگ‌ترین مقدار) ستون 'Flight Distance' را محاسبه و در `max_val` ذخیره می‌کند.

اعمال تبدیل مقیاس کمینه-بیشینه برای 'Flight Distance' =

- final_dataset['Flight Distance'] = (final_dataset['Flight Distance'] - min_val) / (max_val - min_val):

این خط هر مقدار در ستون 'Flight Distance' را با استفاده از فرمول کمینه-بیشینه مقیاس‌بندی می‌کند. مقیاس‌بندی شده‌ها در همان ستون ذخیره می‌شوند.

تکرار روند برای ستون 'Age'=

همان مراحل برای ستون Age تکرار می‌شود. ابتدا کمینه و بیشینه محاسبه شده و سپس فرمول مقیاس‌بندی به کار گرفته می‌شود.

این روش مقیاس‌بندی به یکنواخت سازی مقیاس‌های مختلف داده‌ها کمک می‌کند، بدین معنی که تمام ویژگی‌ها در یک مقیاس استاندارد و یکسان قرار می‌گیرند.

شبکه عصبی نیاز به داده‌هایی با مقیاس‌های یکنواخت دارند تا بتوانند بهتر کار کنند. مقیاس‌بندی کمینه-بیشینه به تسهیل این فرایند کمک می‌کند.

در بخش بعدی شبکه عصبی را تشکیل میدهم . به کمک ان رضایت از پرواز رو میتوانیم بسنجیم.

شبکه عصبی قادر است الگوها را در داده ها شناسایی کند وانها را یاد بگیرد. و برای مسئله های که پیچیده هستند و قواعد واضحی ندارد میتواند کمک کننده باشد.

1 .تعریف کلاس شبکه عصبی:

class NeuralNetwork : یک کلاس برای ایجاد شبکه عصبی تعریف شده است.

2 .متد سازنده:(Constructor)

def __init__(self, layer_sizes): این متد سازنده کلاس، اندازه هر لایه از شبکه عصبی را به عنوان ورودی دریافت می‌کند.

3 .مقداردهی اولیه پارامترها:

def initialize_parameters(self) : این متد وزن‌ها و بایاس‌ها را برای هر لایه از شبکه به صورت تصادفی مقداردهی اولیه می‌کند.

4 .توابع فعال‌سازی:

def relu(self, Z) : تابع فعال‌سازی ReLU

def sigmoid(self, Z) : تابع فعال‌سازی sigmoid

5. Forward Pass:

def forward(self, X) : این متد داده‌ها را از طریق شبکه عصبی به جلو (از ورودی تا خروجی) هدایت می‌کند.

6. Backward Pass و به‌روزرسانی پارامترها:

def backward(self, activations, Y) : محاسبه گرادیان‌ها برای یادگیری شبکه.

def update_parameters(self, grads, learning_rate) : به‌روزرسانی پارامترها بر اساس گرادیان‌های محاسبه شده.

7 .آموزش مدل:

def train(self, X, Y, learning_rate, iterations):آموزش شبکه عصبی با استفاده از داده‌های آموزشی.

8. پیش‌بینی:

def predict(self, X) : انجام پیش‌بینی با استفاده از مدل آموزش دیده.

9. محاسبه دقت:

def accuracy(self, Y_pred, Y_test) : محاسبه دقت مدل.

10. تابع loss و ترسیم نمودارloss :

def cross_entropy_loss(self, Y, Y_pred) : محاسبه loss با استفاده از تابع هزینه-Cross Entropy.

def plot_loss(self) : ترسیم نمودار loss در طول دوره‌های آموزش.

در بخش بعدی آموزش یک شبکه عصبی برای پیش‌بینی رضایت از پرواز است.

این کارها به منظور ساخت و آموزش یک مدل شبکه عصبی برای پیش‌بینی رضایت مسافران از تجربه پرواز انجام می‌شود. هدف از این کار، توسعه یک مدل پیش‌بینی‌کننده دقیق است که بتواند از داده‌های موجود برای تصمیم‌گیری‌های بهتر و بهینه‌سازی خدمات استفاده شود.

1. انتخاب نمونه‌ها:

no_samples = 2000: تعداد نمونه‌هایی که می‌خواهیم برای آموزش و تست شبکه عصبی استفاده کنیم.

Sample نمونه تصادفی ای که از مجموعه داده‌ها (final_dataset) انتخاب می‌شود.

2. تقسیم داده‌ها به دو بخش آموزش و تست:

train_size = int(0.8 * no_samples) 80 درصد از نمونه‌ها برای آموزش و 20 درصد برای تست اختصاص داده می‌شود.

Y_train و Y_test: برچسب‌های مربوط به رضایت از پرواز برای داده‌های آموزشی و تست استخراج می‌شوند.

sample.drop(columns=['satisfaction'], inplace=True): ستون رضایت از داده‌های ویژگی‌ها حذف می‌شود.

3. آماده‌سازی ورودی‌ها و خروجی‌های شبکه عصبی:

- X = sample.to_numpy: ویژگی‌ها به فرمت آرایه نامپای (NumPy) تبدیل می‌شوند.
- X_train و X_test: داده‌های ورودی برای آموزش و تست تقسیم می‌شوند.

4. تعریف معماری شبکه عصبی:

input_size, hidden_size, output_size: تعداد نورون‌ها در لایه‌های ورودی، پنهان، و خروجی تعیین می‌شود.

5. آموزش شبکه عصبی:

net = NeuralNetwork([input_size, hidden_size, output_size]): شبکه عصبی با معماری مشخص شده ایجاد می‌شود.

net.train: شبکه با استفاده از داده‌های آموزشی آموزش داده می‌شود.

6. ارزیابی شبکه:

Y_pred = net.predict(X_test): شبکه برای پیش‌بینی رضایت از پرواز روی داده‌های تست اجرا می‌شود.

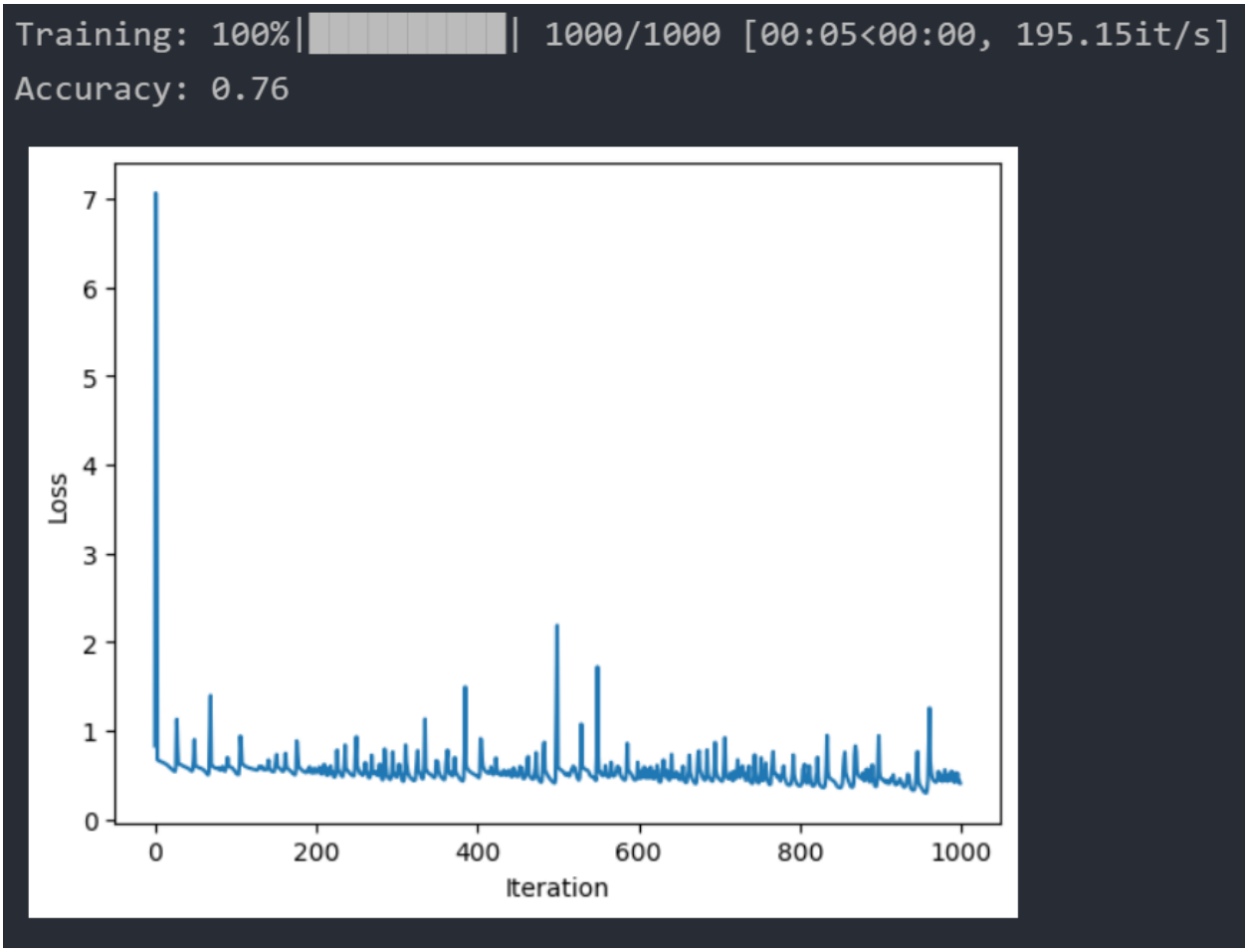
accuracy = net.accuracy(Y_pred, Y_test): دقت شبکه محاسبه می‌شود.

7. نمایش نتایج:

net.plot_loss: نمودار تغییرات خطا در طول آموزش نمایش داده می‌شود.

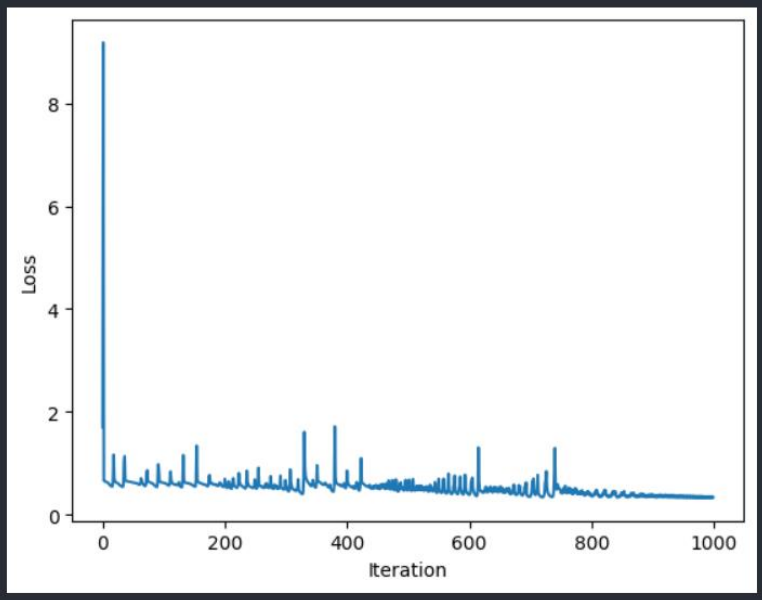
- نتایج پیش‌بینی‌ها به همراه برچسب‌های واقعی در یک فایل CSV ذخیره می‌شوند.

وقتی با 2000 نمونه train رو انجام دادم accuracy و نمودار loss در کل فرایند اموزش به صورت زیر شد :



وقتی با 5000 نمونه train رو انجام دادم accuracy و نمودار loss در کل فرایند اموزش به صورت زیر شد :

```
Training: 100%|██████████| 1000/1000 [00:11<00:00, 84.23it/s]
Accuracy: 0.89
Feature importances: [0.008 0.005 0.035 0.02  0.106 0.02  0.042 0.003 0.102 0.195]
```



با توجه به اختلاف accuracy در دو بخش اخر متوجه میشویم که افزایش تعداد نمونه باعث افزایش accuracy شد پس شبکه عصبی با دیدن دیتاها بیشتر بهتر اموزش دید و پاسخ بهتری هم بهمون خواهد داد .

در بخش بعدی فیچرها را بر اساس اهمیت و تاثیرگذاری شان مرتب کردم. برای رسیدن به این خواست مسئله از انتروپی استفاده کردم. و تاثیرگذاری هر ستون بر اساس مقدار اطلاعاتی هست که به ما میدهد.

در کد انتروپی و انتروپی شرطی پیاده سازی شده است. پس از اعمال فرمولهای لازم نمودار زیر به دست امد :

1. Joint Entropy:

$$H(X,Y) = -\sum\sum P(x,y)\log_2 P(x,y)$$

2. Entropy:

$$H(X) = -\sum_{i=1}^{n} P(x_i)\log P(x_i)$$

3. Information Gain:

$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$