بسم الله الرحمن الرحیم

دانشجو: محمدرضا جنیدی جعفری ۹۹۲۵۲۵۳

درس مبانی سیستم های هوشمند

استاد: دکتر مهدی علیاری

مینی پروژه اول

لینک مخزن گیت هاب / گوگل کولب سوال یک / گوگل کولب سوال دو

مهم: در تمامی مراحل کد نویسی، مقدار random_state = 53 قرار گرفته شده است.

۱-

۱-۱-

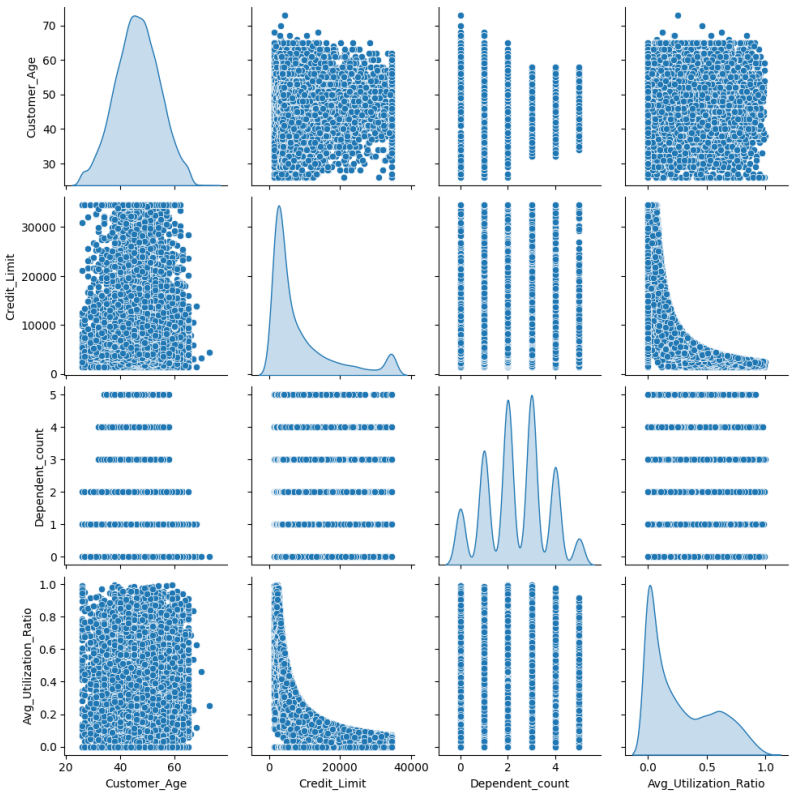
این دیتاست بر اساس ادعایی که نویسنده در سایت ارائه کرده است، به منظور پیشبینی اینکه کدام یک از مشتریان خدمات خود را لغو خواهند کرد ساخته شده است تا بتوانیم خدمات بهتری را به مشتریان ارائه کنیم.

این داده مجموعه حدود **10,000** نمونه دارد که شامل سن، حقوق، وضعیت تاهل و وِیژگی های دیگر است. چالش حال حاضر این مجموعه آن است که تنها 16.07% مشتریان خدمات خود را لغو کرده اند که این موضوع آموزش مدل را کمی سخت می کند.

این داده مجموعه به صورت کلی 23 ستون دارد که ستون اول شامل ID هر فرد است و نمی شود آنرا به عنوان یک ویژگی در نظر گرفت. با در نظر گرفتن دو ستون آخر 22 ویژگی در این مجموعه وجود دارد.

۱-۲-

با انتخاب ۴ ویژگی که در تصویر مشخص شده اند، و با استفاده از دستور seaborn.pairplot نمودار زیر بدست آمد:



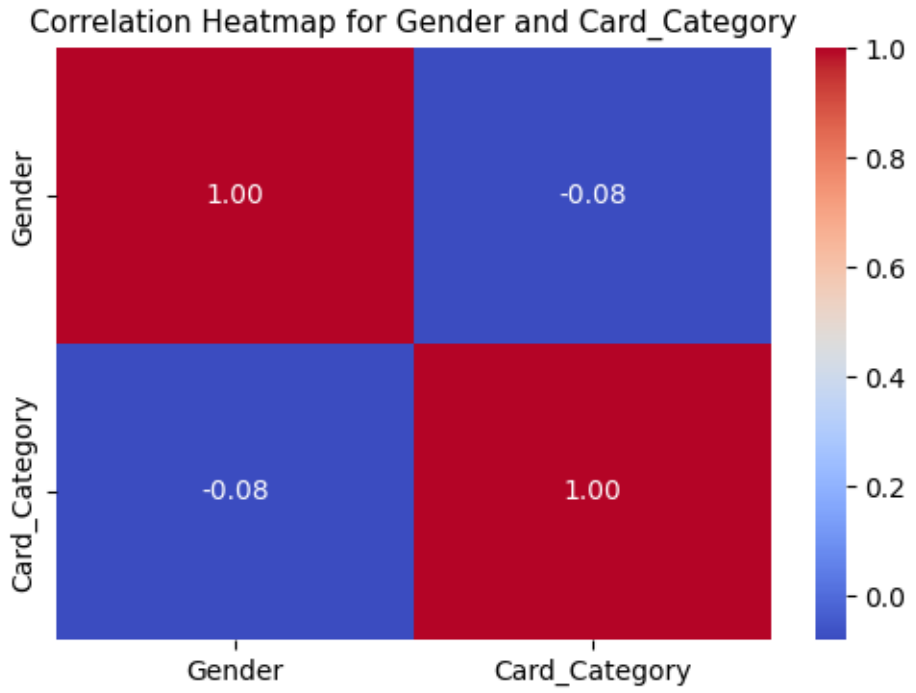
۱-۳-

در این مرحله، برای درک راحت تر مدل و کار آسان تر با داده ها، ارزش های متنی را به صورت عددی تبدیل می کنیم.
Blue = 0, Gold = 0, Silver = 2

M = 0, F = 1

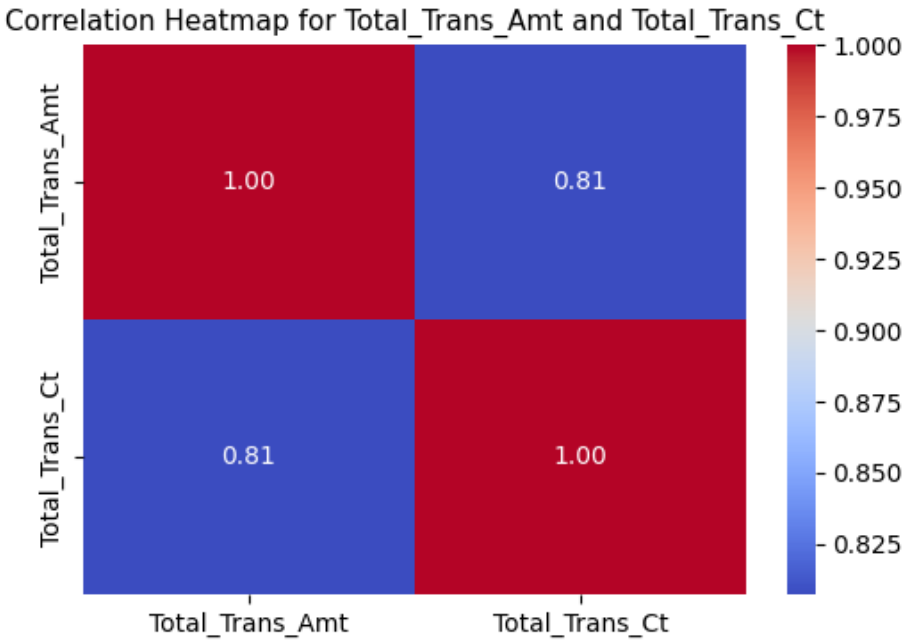
این امکان، با دستور map انجام شده است.

همبستگی بین دو ویژگی Gender و Card_Category که Categorical هستند به صورت زیر مشاهده شد:



همانطور که مشاهده می شود، عدد 0.08- به عنوان هبستگی، نشان از آن دارد که این دو ویژگی تقریبا هیچ اثری روی یکدیگر ندارند.

حالا برای دو ویژگی Total_Trans_Amt و Total_Trans_Ct که ویژگی های پیوسته هستند این کار را تکرار می کنیم:



عدد 0.81 نشان دهنده ی رابطه قوی بین دو ویژگی است و تغییرات هم به یکدیگر تاثیر می گذارند.

۴-۱-

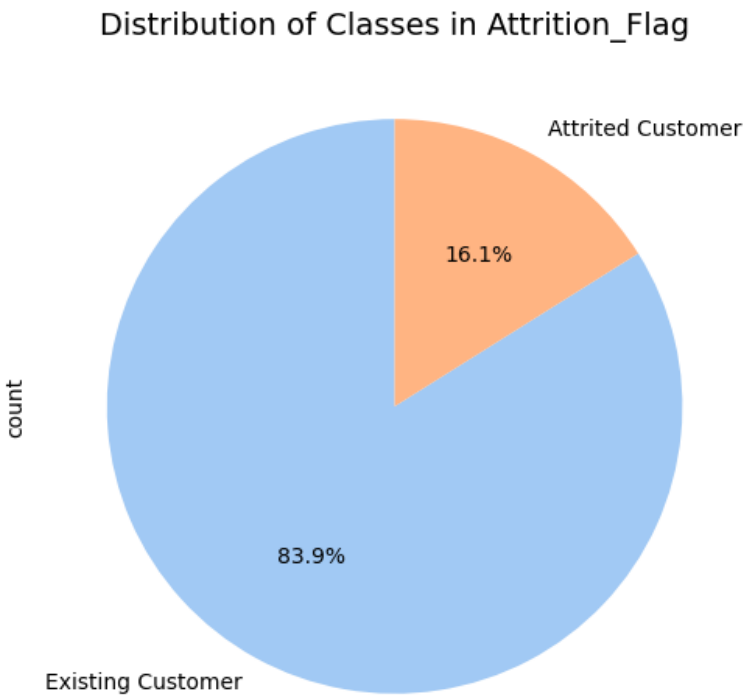
```
# Check for NaN values in the dataset
nan_rows = df[df.isna().any(axis=1)]

# Display rows with NaN values (if any exist)
if not nan_rows.empty:
    print("Rows with NaN values:")
    print(nan_rows)

    print("\nRow numbers with NaN values:")
    print(nan_rows.index.tolist())

    df_cleaned = df.dropna()
    print("\nDataset after removing rows with NaN values:")
    print(df_cleaned)
else:
    print("No NaN values found in the dataset.")
```

کد بالا نشان می دهد که چه مقدار داده NaN در مجموعه وجود دارد که خروجی آن به صورت زیر است:

No NaN values found in the dataset.

۵-۱-

با توجه به قسمت ۵-۱ کد در کولب، این ویژگی دارای دو کلاس ['Existing Customer' 'Attrited Customer'] است و تعداد آنها 2 است. نمودار pie توزیع این کلاس ها به صورت زیر است:



همانطور که مشاهده می شود، این ویژگی دارای عدم تعادل است که می تواند تاثیرات منفی روی آموزش مدل بگذارد. به این دلیل که در این کلاس توزیع کلاس Existing خیلی بیشتر از Attrited است و این یعنی، مدل یکی از کلاس ها را بهتر یاد خواهد گرفت.

می توان از الگوریتم SMOTE استفاده کرد:

الگوریتم SMOTE یک روش محبوب برای مقابله با مشکل داده‌های نامتعادل است. این الگوریتم با ایجاد نمونه‌های مصنوعی از کلاس اقلیت، تعداد نمونه‌های این کلاس را افزایش می‌دهد و به این ترتیب تعادل بین کلاس‌ها را برقرار می‌کند.

نحوه کار SMOTE:

- یافتن همسایگان نزدیک :برای هر نمونه از کلاس اقلیت، kنزدیک‌ترین همسایه آن در فضای ویژگی‌ها پیدا می‌شود.
- یجاد نمونه‌های مصنوعی :بین هر نمونه از کلاس اقلیت و همسایگان آن، نمونه‌های مصنوعی جدیدی ایجاد می‌شود. این نمونه‌های جدید در امتداد خطی که نمونه اصلی را به همسایه آن متصل می‌کند، ایجاد می‌شوند.
- تکرار فرایند :این فرایند برای تمام نمونه‌های کلاس اقلیت تکرار می‌شود تا تعداد نمونه‌های این کلاس به تعداد دلخواه برسد.

مزایای:SMOTE

- افزایش تعداد نمونه‌های کلاس اقلیت SMOTE :به طور مستقیم تعداد نمونه‌های کلاس اقلیت را افزایش می‌دهد و به این ترتیب تعادل بین کلاس‌ها را برقرار می‌کند.
- سادگی :این الگوریتم به سادگی قابل پیاده‌سازی است و نیازی به تنظیم پارامترهای پیچیده ندارد.
- کارایی SMOTE :معمولاً کارایی بالایی دارد و می‌تواند بر روی مجموعه داده‌های بزرگ اعمال شود.

بهتر است الگوریتم SMOTE را **قبل** از تقسیم بندی داده‌ها به بخش‌های آموزش و آزمون اعمال کنیم. زیرا:

- **قبل از تقسیم بندی :**تضمین توزیع متعادل در هر دو بخش و جلوگیری از نشت اطلاعات .
- **بعد از تقسیم بندی :**ممکن است باعث نشت اطلاعات شود و ارزیابی مدل را تحت تاثیر قرار دهد.

۶-۱-

بدون متعادل کردن داده ها:

داده ها به صورت 20% 20% ,40% برای آموزش، ارزیابی و تست تقسیم بندی شدند و از مدل آماده RandomForestاستفاده شده است.

Class Distribution in Training Set:

Attrition_Flag

1 0.839368

0 0.160632

Name: proportion, dtype: float64

Class Distribution in Validation Set:

Attrition_Flag

1 0.839506

0 0.160494

Name: proportion, dtype: float64

Class Distribution in Test Set:

Attrition_Flag

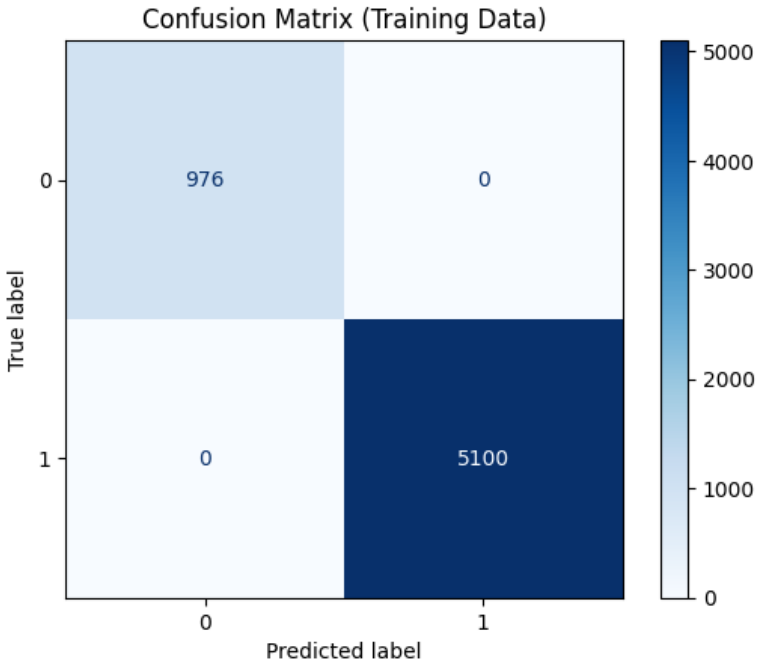
1 0.839092

0 0.160908

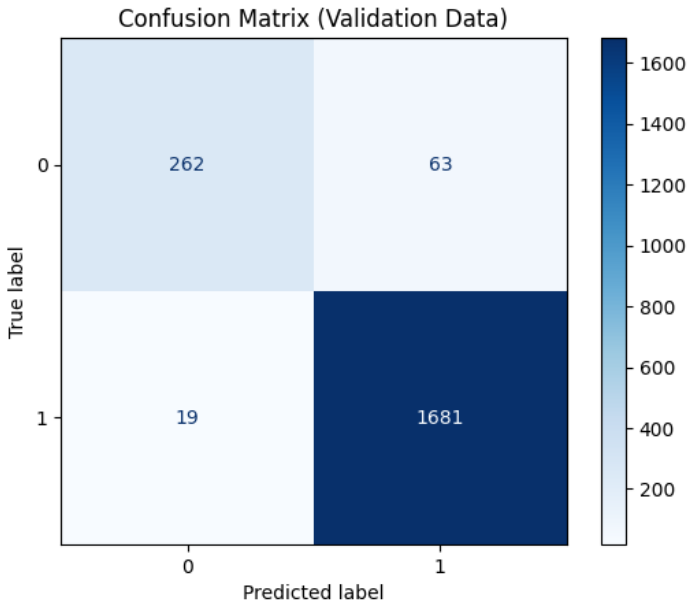
Name: proportion, dtype: float64

نتایج آموزش، ارزیابی و تست به صورت زیر خواهد بود:

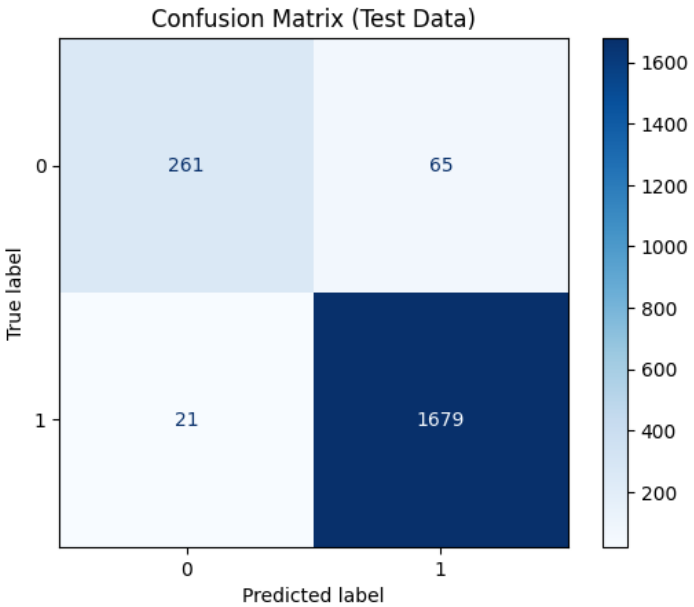
```
Classification Report (Training Data):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       976
           1       1.00      1.00      1.00      5100

    accuracy                           1.00      6076
   macro avg       1.00      1.00      1.00      6076
weighted avg       1.00      1.00      1.00      6076
```




```
Classification Report (Validation Data):
              precision    recall  f1-score   support

           0       0.93      0.81      0.86       325
           1       0.96      0.99      0.98      1700

    accuracy                           0.96      2025
   macro avg       0.95      0.90      0.92      2025
weighted avg       0.96      0.96      0.96      2025
```



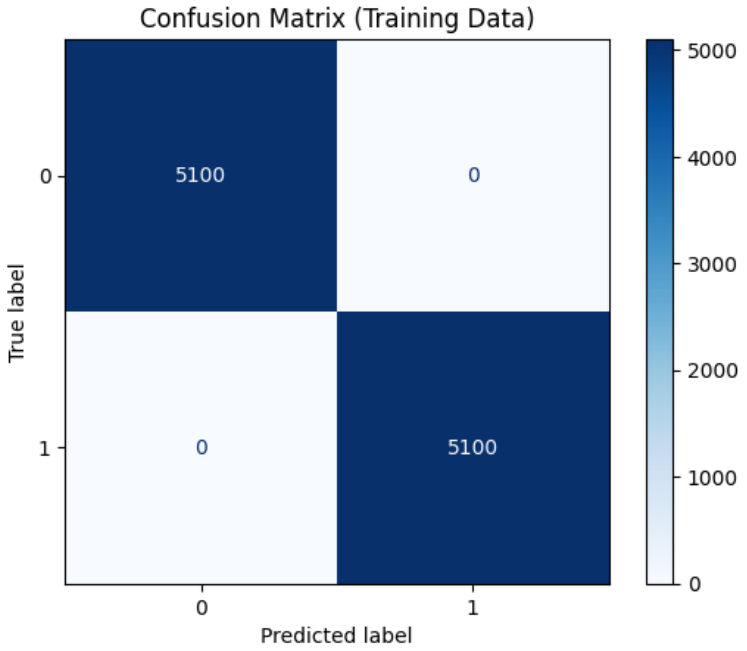

```
Accuracy on Test Data: 0.96

Classification Report (Test Data):
              precision    recall  f1-score   support

           0       0.93      0.80      0.86       326
           1       0.96      0.99      0.98      1700

    accuracy                           0.96      2026
   macro avg       0.94      0.89      0.92      2026
weighted avg       0.96      0.96      0.96      2026
```



حالا در این قسمت، از روش SMOTE برای متعادل کردن دیتا روی ۸۵۰۰ استفاده می کنیم.

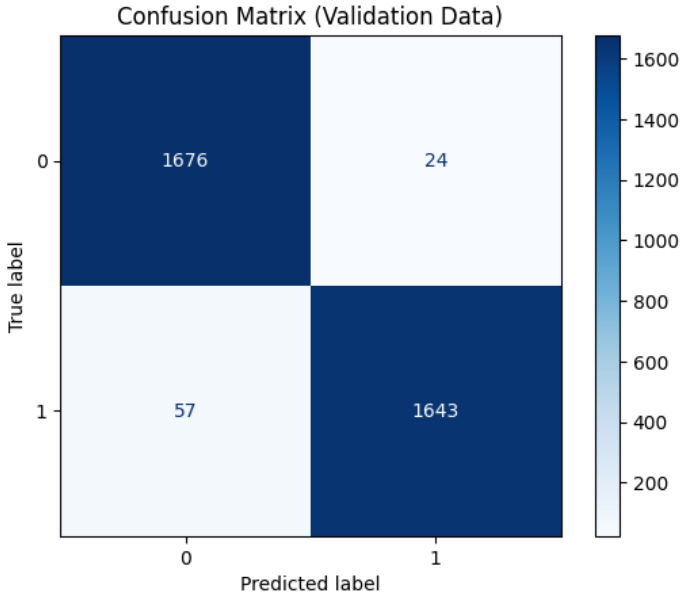
Classification Report (Training Data):

	precision	recall	f1-score	support
0	1.00	1.00	1.00	5100
1	1.00	1.00	1.00	5100
accuracy			1.00	10200
macro avg	1.00	1.00	1.00	10200
weighted avg	1.00	1.00	1.00	10200

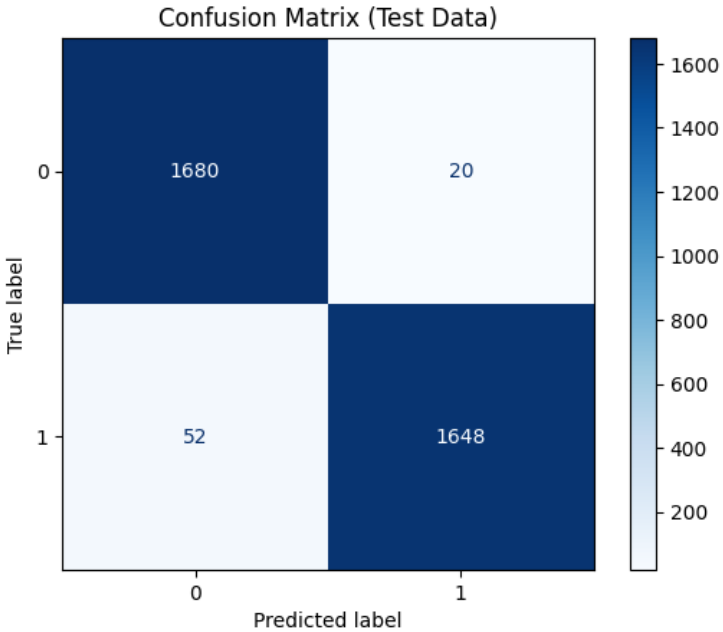


Classification Report (Validation Data):

	precision	recall	f1-score	support
0	0.97	0.99	0.98	1700
1	0.99	0.97	0.98	1700
accuracy			0.98	3400
macro avg	0.98	0.98	0.98	3400
weighted avg	0.98	0.98	0.98	3400




```
Accuracy on Test Data: 0.98

Classification Report (Test Data):
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1700
           1       0.99      0.97      0.98      1700

    accuracy                           0.98      3400
   macro avg       0.98      0.98      0.98      3400
weighted avg       0.98      0.98      0.98      3400
```

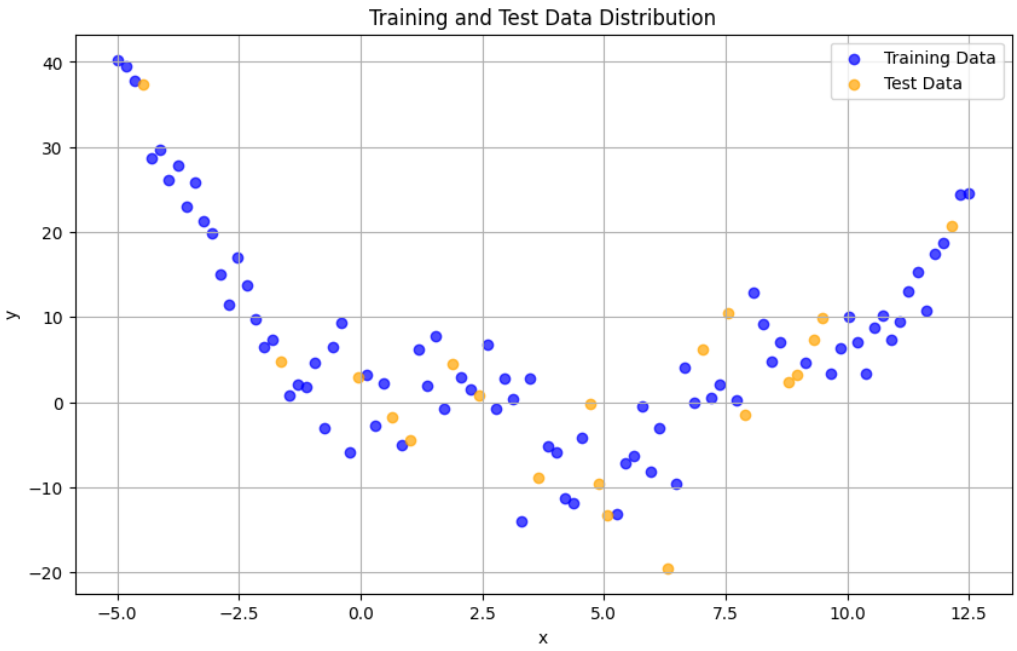


همانطور که مشاهده می شود، متعادل کردن داده ها روی نتایج تست تاثیر مثبت گذاشته است و نتیجه را 2% بهتر کرده است.

-۲

-۲-۱

با توجه به کد موجود در کولب داده ها را فراخواندیم و داده ها را با نسبت یک به چهار به تست و آموزش تقسیم کردیم:



. -۲-۲

در مدل‌های رگرسیونی، هدف پیش‌بینی یک مقدار پیوسته است. برای ارزیابی دقت این پیش‌بینی‌ها، از معیارهای مختلفی استفاده می‌شود. در ادامه، سه معیار پرکاربرد را معرفی می‌کنیم:

۱- میانگین مربعات خطا(Mean Squared Error - MSE)

- **تعریف** MSE **:**برابر است با میانگین مربعات تفاوت بین مقادیر پیش‌بینی شده توسط مدل و مقادیر واقعی.
- **فرمول:**

MSE = (1/n) * Σ(y_i - y_hat_i)^2

که در آن:

- n:تعداد نمونه‌ها
- y_i:مقدار واقعی متغیر وابسته برای نمونهi
- y_hat_i:مقدار پیش‌بینی شده متغیر وابسته برای نمونهi

- **تفسیر MSE :**هرچه کمتر باشد، نشان‌دهنده دقت بیشتر مدل است. با این حال، به دلیل مجذور کردن خطاها، خطاهای بزرگ تأثیر بیشتری بر MSE دارند.

۲- میانگین قدر مطلق خطا(Mean Absolute Error - MAE)

- **تعریف MAE :**برابر است با میانگین قدر مطلق تفاوت بین مقادیر پیش‌بینی شده و مقادیر واقعی.
- **فرمول:**

$$MAE = (1/n) * \Sigma|y_i - y\_hat_i|$$

- **تفسیر MAE :**نسبت به MSE به تغییرات شدید حساسیت کمتری دارد و تفسیر آن ساده‌تر است .MAE نشان‌دهنده میانگین خطای مطلق پیش‌بینی‌ها است.

۳- R-squared (R^2)

- **تعریف R-squared :**نشان می‌دهد که چه مقدار از تغییرات متغیر وابسته توسط مدل توضیح داده شده است.
- **تفسیر R-squared :**بین ۰ تا ۱ متغیر است. هرچه R-squared به ۱ نزدیک‌تر باشد، مدل بهتر است و تغییرات متغیر وابسته را بهتر توضیح می‌دهد .R-squared برابر با صفر نشان می‌دهد که مدل هیچ توانایی در توضیح تغییرات متغیر وابسته ندارد.

۳-۲-

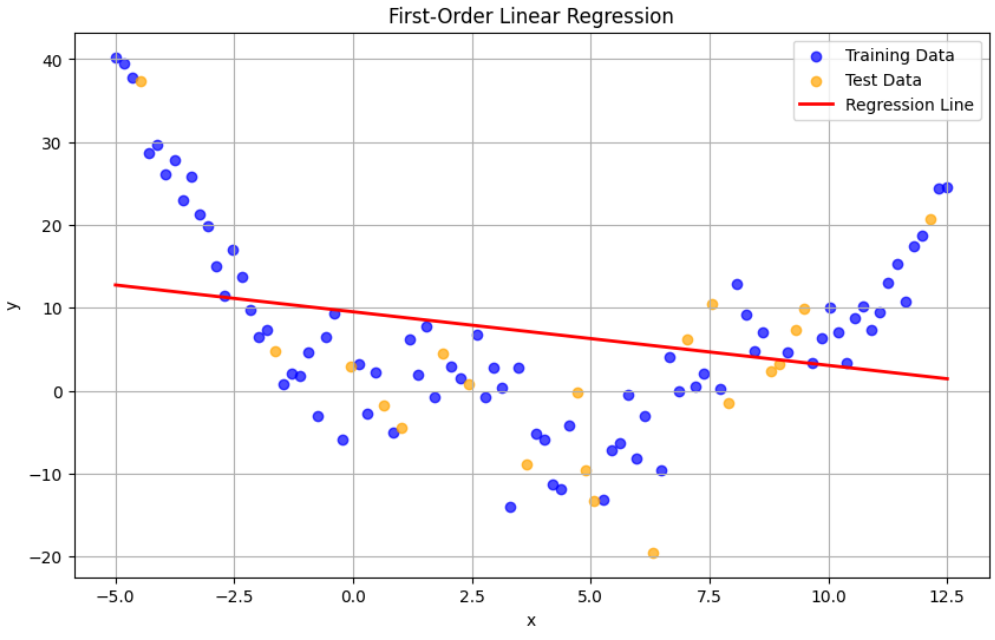
یک معادله خطی درجه یک به صورت زیر است:

$$y = mx + b$$

$$m = \frac{Cov(x,y)}{Var(x)}$$

$$b = \bar{y} - m\,\bar{x}$$

معادلات بالا را در سلول کولب می نویسیم و مدل رگرسیون خطی درجه یک را تولید می کنیم:



همانطور که مشاهده می شود و از نتایج زیر مشخص است، مدل رگرسیون خطی درجه یک نمی تواند مدل مناسبی برای این داده باشد:

Slope (m): -0.6468773697252911

Intercept (b): 9.5195587566984

MSE on Training Data: 129.7546407700454

MSE on Test Data: 155.7522683264296

۴-۲-

در این مرحله، در هر دور تعداد نمونه های داده ی آموزش را افزایش می دهیم:

خطای آموزش:

- در ابتدا، خطای آموزشی بسیار کم خواهد بود زیرا مدل کاملاً با داده های آموزشی کوچک مطابقت دارد.
- با افزایش تعداد نمونه های آموزشی، خطای آموزشی ممکن است کمی افزایش یابد، اما با تعمیم بهتر مدل تثبیت خواهد شد.

خطای تست:

- خطای تست زمانی که فقط از چند نمونه آموزشی استفاده می شود زیاد خواهد بود، زیرا مدل فاقد برازش است.
- با افزایش تعداد نمونه های آموزشی، خطای تست کاهش می یابد زیرا مدل بهتر تعمیم می یابد.

-۲-۵

افزایش حجم داده‌های آموزشی می‌تواند با کمک به مدل برای درک بهتر الگوهای موجود در داده‌ها و تعمیم به نمونه‌های ندیده، عملکرد آن را بهبود بخشد .با این حال، محدودیت‌هایی در میزان بهبودی که داده‌های اضافی می‌توانند فراهم کنند، وجود دارد:

- اگر معماری یا فرضیه مدل بیش از حد ساده باشد (مثلاً یک چندجمله‌ای درجه اول برای داده‌های بسیار غیرخطی)، صرف نظر از مقدار داده، نمی‌تواند پیچیدگی را درک کند.
- اگر مدل قبلاً الگوهای ذاتی داده‌ها را درک کرده باشد، افزودن داده‌های بیشتر بازده کاهشی خواهد داشت.

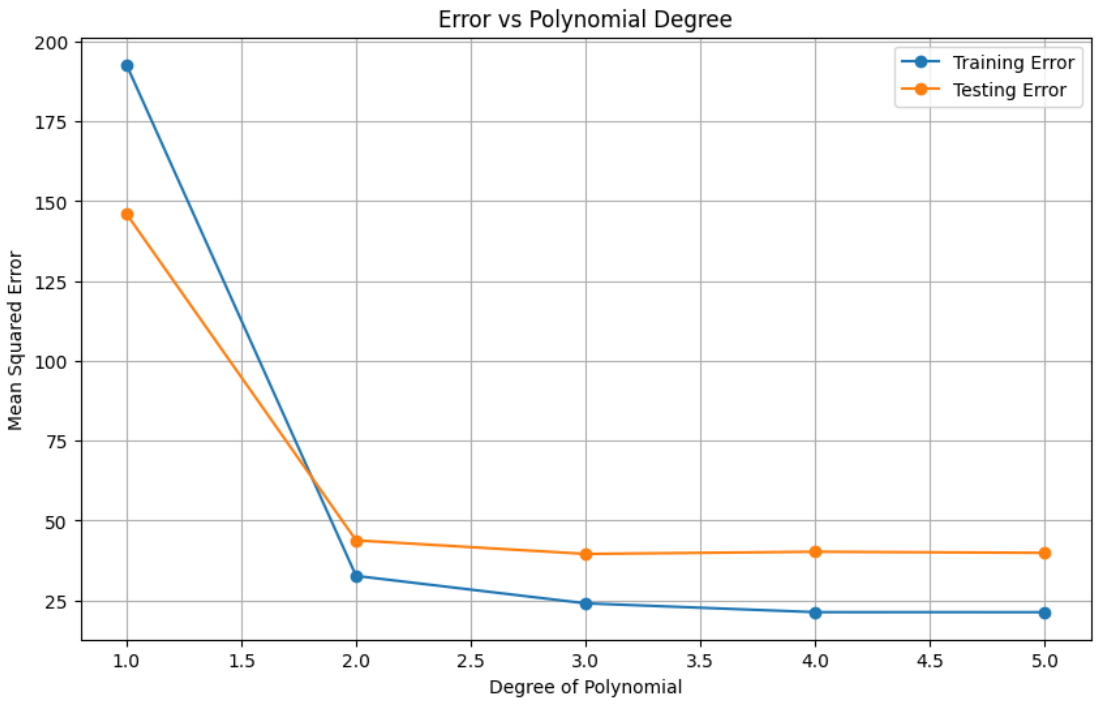
خطای انسانی نشان‌دهنده نویز ذاتی یا خطای غیرقابل کاهش در داده‌ها است. این بهترین سناریو است .خطای مدل دارای سه مؤلفه است:

- بایاس: خطای ناشی از فرض‌های نادرست یا کم‌فیت شدن.
- واریانس: خطای ناشی از حساسیت به نوسانات در داده‌های آموزشی.
- خطای غیرقابل کاهش: نویز یا تصادفی بودن در داده‌ها که نمی‌توان آن را با هیچ مدلی حذف کرد .حتی با داده‌های بی‌نهایت، خطای مدل نمی‌تواند کمتر از خطای غیرقابل کاهش (نمایش داده شده توسط خطای انسانی) شود.

اگر خطای فعلی به دلیل بایاس (کم‌فیت شدن) باشد، افزودن داده‌های بیشتر به تنهایی آن را کاهش نخواهد داد. پیچیدگی مدل باید افزایش یابد (مثلاً با استفاده از چندجمله‌ای‌های با درجه بالاتر یا مدل‌های پیشرفته‌تر). اگر خطای به دلیل واریانس (بیش‌فیت شدن) باشد و مدل با تعمیم مشکل داشته باشد، داده‌های بیشتر می‌تواند به کاهش واریانس و بهبود عملکرد کمک کند. اگر خطا نزدیک به خطای غیرقابل کاهش (خطای انسانی) باشد، هیچ مقدار داده‌ای نمی‌تواند آن را بیشتر کاهش دهد.

-۲-۶

مدل را از درجه یک تا درجه ۵ پیاده سازی می کنیم:



به طور دقیق تر داریم:

Degree	Training Error	Testing Error
1	192.7595	146.0427
2	32.67154	43.80528
3	24.08366	39.5568
4	21.32254	40.23817
5	21.31153	39.89385

درجه ۳ بهترین است زیرا خطای تست را به حداقل می رساند (۳۹.۵۶). این نشان می‌دهد که یک چند جمله‌ای مکعبی تعادل بهینه را بین بایاس (زیاد برازش) و واریانس (بیش از حد برازش) برای این مجموعه داده ایجاد می‌کند.

-۲-۷

۱. **رگرسیون خطی**

مدل رگرسیون خطی رابطه بین ویژگی‌های ورودی (X) و متغیر هدف (y) را با برازش یک معادله خطی مدل‌سازی می‌کند $y=\beta 0+\beta 1x1+\cdots+\beta nxn$:این مدل مجموع مربعات باقیمانده (تفاوت بین مقادیر مشاهده‌شده و پیش‌بینی‌شده) را به حداقل می‌رساند.

چه زمانی استفاده شود : برای داده‌هایی با روابط خطی بین ویژگی‌ها و هدف بهترین است.

مزایا :پیاده‌سازی و تفسیر ساده.

معایب : با روابط غیرخطی مشکل دارد.

۲. **درخت تصمیم**

این الگوریتم داده‌ها را بر اساس آستانه‌های ویژگی به مناطق تقسیم می‌کند و یک مدل درختی ایجاد می‌کند. پیش‌بینی‌ها با میانگین‌گیری متغیر هدف در گره‌های برگ انجام می‌شوند .این مدل روابط غیرخطی را به خوبی درک می‌کند.

چه زمانی استفاده شود : برای مجموعه داده‌هایی با روابط غیرخطی و تعامل بین ویژگی‌ها مفید است.

مزایا :هندلینگ روابط غیرخطی .می‌تواند تعامل بین متغیرها را مدل‌سازی کند.

معایب :مستعد بیش‌برازش، به ویژه با درختان عمیق.

۳. **جنگل تصادفی**

یک روش آنسامبل که چندین درخت تصمیم می‌سازد و پیش‌بینی‌های آن‌ها را ترکیب می‌کند (میانگین‌گیری برای رگرسیون) .در مقایسه با درختان تصمیم فردی، بیش‌برازش را کاهش می‌دهد.

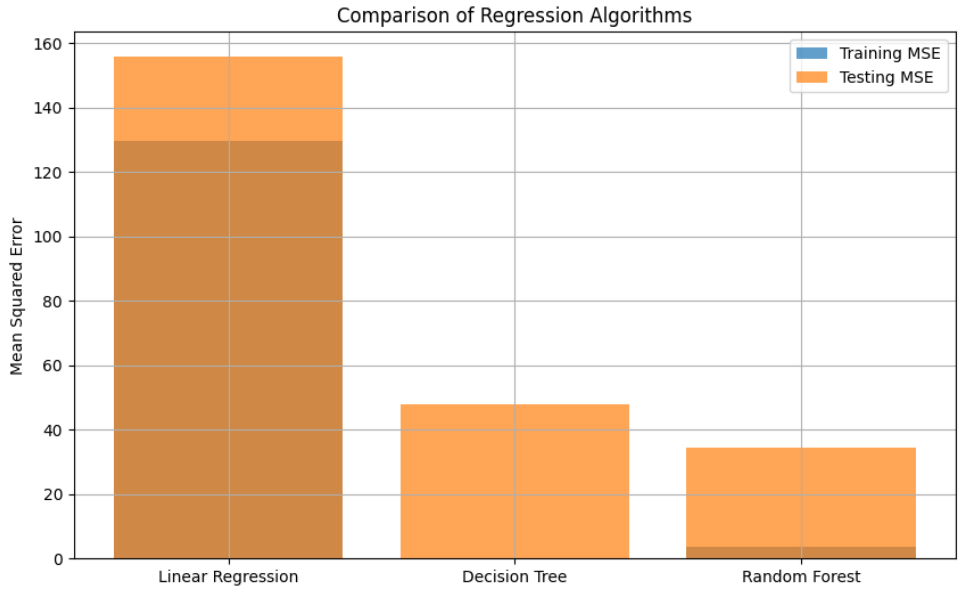
چه زمانی استفاده شود: برای روابط خطی و غیرخطی مؤثر است.

مزایا: مقاوم در برابر بیش‌برازش .هندلینگ خوب داده‌های با ابعاد بالا.

معایب: محاسباتی گران برای مجموعه داده‌های بزرگ.

نتایج این سه مدل به صورت زیر بوده است:

Model	Training MSE	Testing MSE	Generalization Ability
Linear Regression	129.7546	155.7523	Poor (Underfitting)
Decision Tree	0	47.9639	Moderate (Overfitting)
Random Forest	3.494616	34.36396	Best (Balanced)



بهترین مدل:

جنگل تصادفی بهترین مدل بین این سه است، زیرا کمترین MSE آزمایشی را بدون بیش‌برازش به دست می‌آورد.

چرا جنگل تصادفی بهترین عملکرد را دارد:

- با ترکیب چندین درخت تصمیم، بیش‌برازش را کاهش می‌دهد (بَگینگ).
- با میانگین‌گیری پیش‌بینی‌ها در بین درختان، تعمیم‌پذیری خوبی دارد.

گام‌های بعدی:

اگر بهبود بیشتری مورد نیاز است:

- تنظیم ابرپارامترها برای جنگل تصادفی (مثلاً تنظیم تعداد درختان، حداکثر عمق).
- افزودن داده‌های بیشتر یا پیش‌پردازش ویژگی‌ها برای بهبود بیشتر عملکرد مدل.