Topics

- quick recap of co-ord geometry for Logistic Regr } ← (earlier)
- ✓ Log-loss (squared-loss) — code
- Logistic regression : optimization
- end to end LR
- code (Churn prediction) — telecom

Prev

Ques → Question tab → track

Chat → Voting & conversation

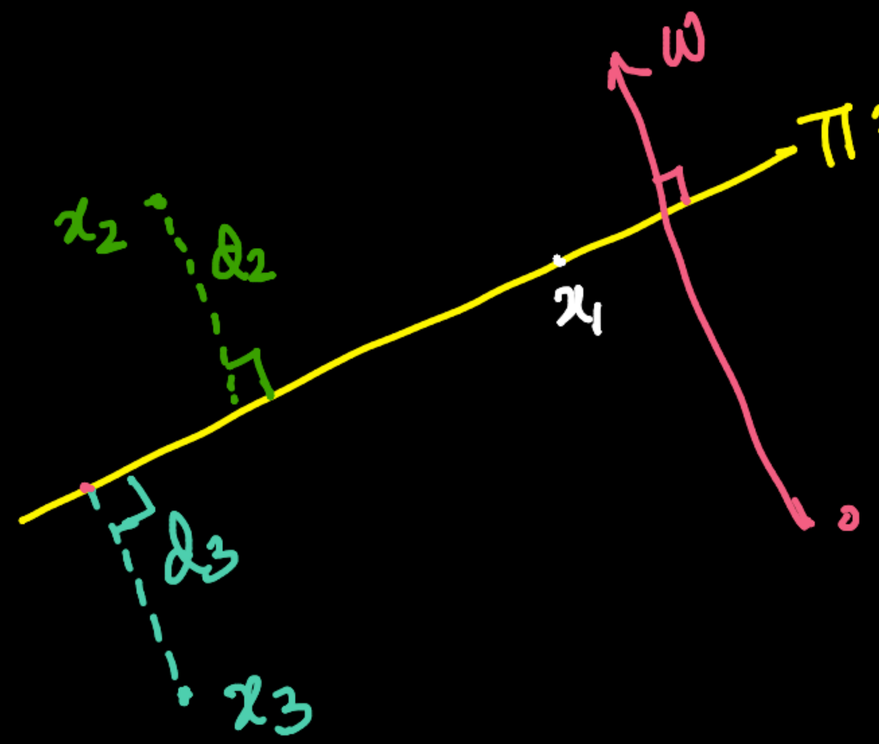
mark answered

Chrome File Edit View History Bookmarks Profiles Tab Window Help 21:10

Recap: Co-ord geom needed for logistic regr:

$d_3 = w^T x_3 + w_0$

(-ve)



$\pi: w^T x + w_0 = 0$

$\boxed{\|w\| = 1}$

$w^T x_1 + w_0 = 0$

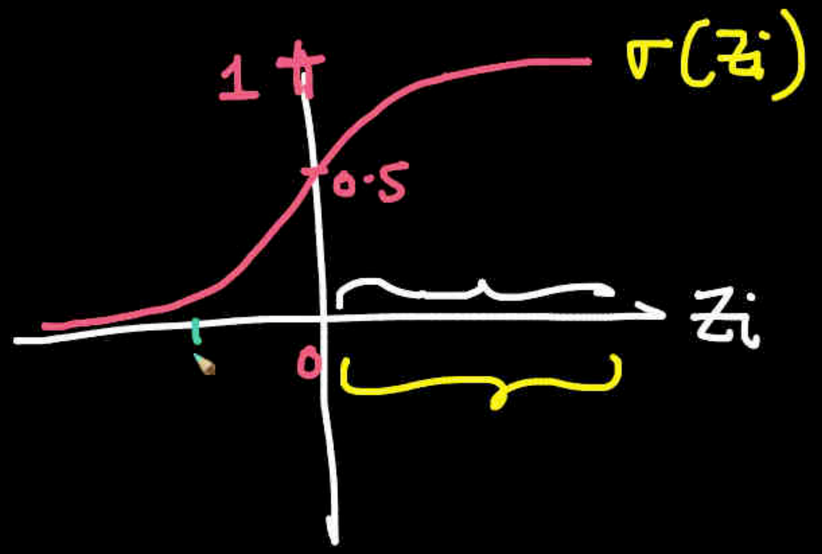
$d_2 = \text{dist}(\pi, x_2)$

(+ve) $d_2 = w^T x_i + w_0$

$$w^T x_i + w_0 = z_i$$

$$\sigma(z_i) = \frac{1}{1+\exp(-z_i)}$$

sigmoid



$$(-\infty, \infty) \longrightarrow (0,1)$$

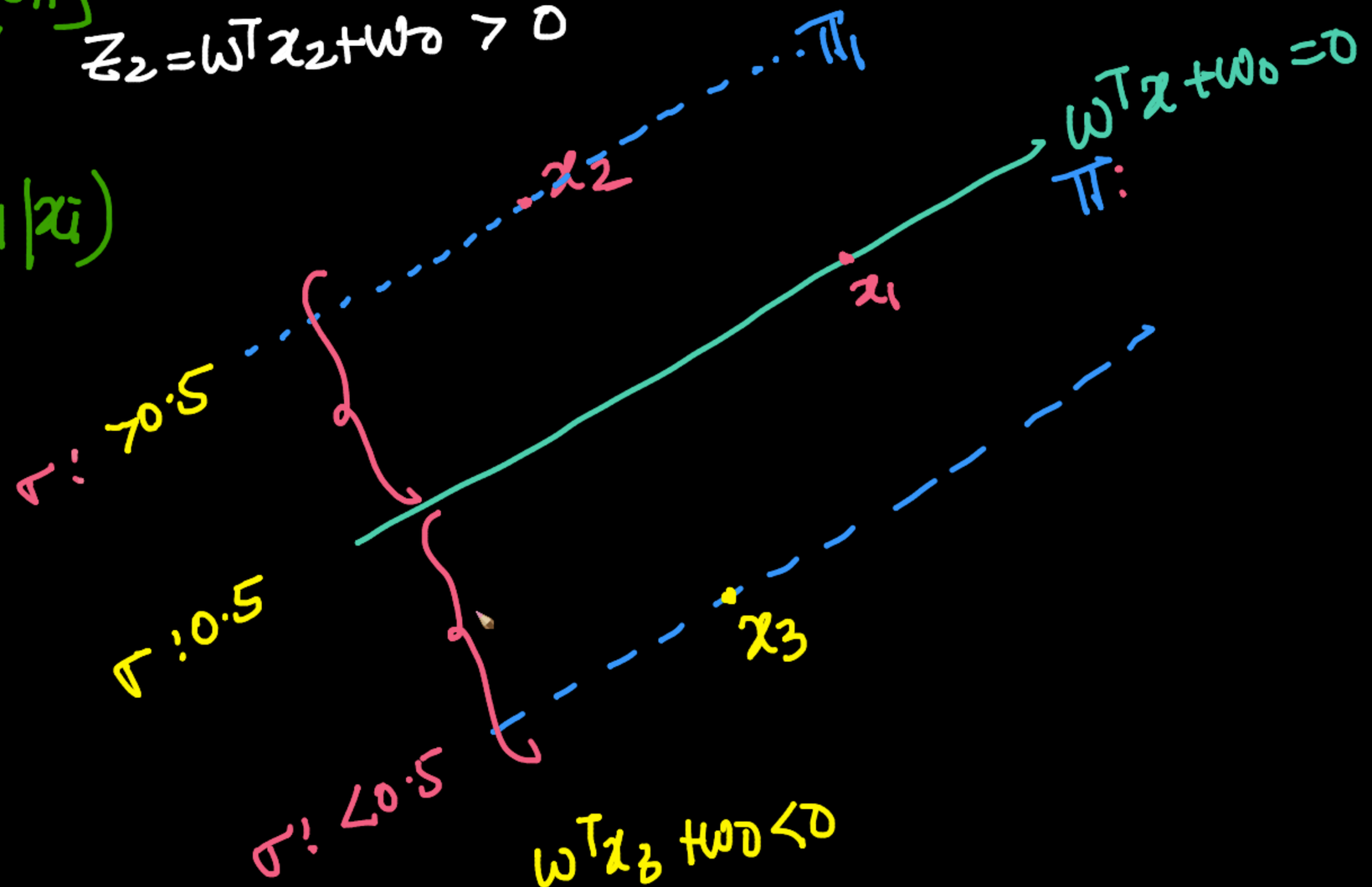
$y_i \in \{0,1\}$

$\downarrow$

$P(y_i = 1 \mid x_i)$

$z_2 = w^T x_2 + w_0 > 0$

$\pi_1 \parallel \pi$



$w^T x_3 + w_0 < 0$

$z_3 < 0$

$\sigma(z_i) = \frac{1}{1+e^{-z_i}}$

$w^T x_1 + w_0 = z_1 = 0$

$\sigma(z_1) = \sigma(0) = 0.5$

✓ will revisit more co-ord geom/LA whenever needed

$w^T x_1 + w_0 > w^T x_2 + w_0$

$z_1 > z_2$

$\sigma(z_1) > \sigma(z_2)$

$(0,1)$

$p(y_i = 1 | x_i)$

$p(y_1 = 1 | x_1) > p(y_2 = 1 | x_2)$

x_1 +ve $\rightarrow 1$

x_2 +ve

$\pi : w^T x + w_0 = 0$

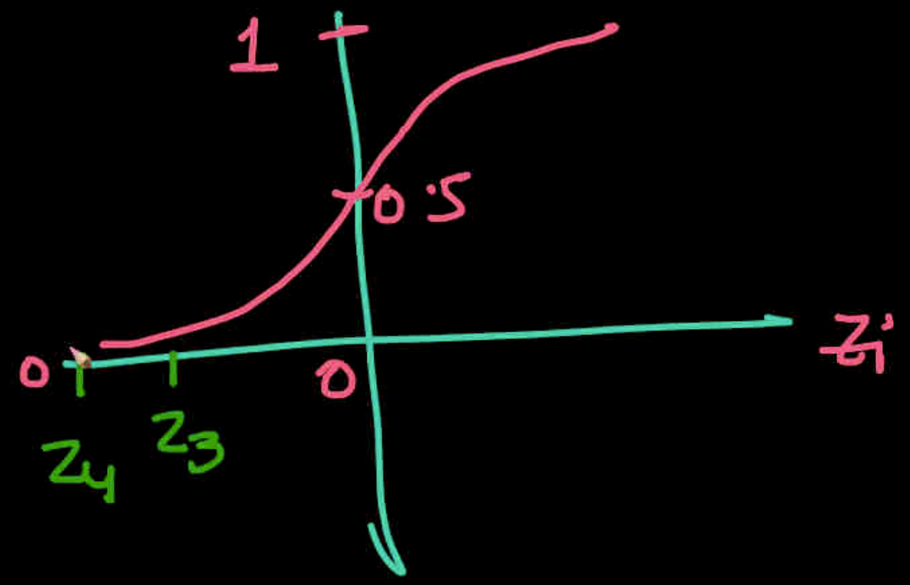
$\|w\| = 1$

$\sigma(z_i)$: monotonically incr fn

$z_i \uparrow \quad \sigma(z_i) \uparrow$

x_3 -ve

x_4 -ve



$$P(y_4=1 \mid x_4) < P(y_3=1 \mid x_3)$$

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | plot(-log(x)) - Google Search | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=DI6u91buPqyH

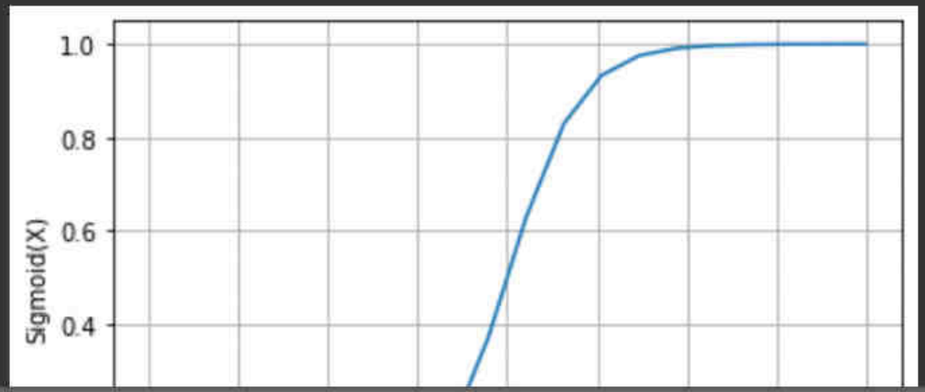
+ Code + Text

RAM Disk

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


def sigmoid(x):
    return 1/(1+np.e**-x)

x = np.linspace(-10, 10, 20)
z = sigmoid(x)
plt.plot(x, z)
plt.xlabel("x")
plt.ylabel("Sigmoid(X)")
plt.grid()
plt.show()
```

$$\frac{1}{1+e^{-z}}$$

-10 to



Logistic Regression.ipynb - Co | plot(-log(x)) - Google Search | plot(-log(1-x)) - Google Searc

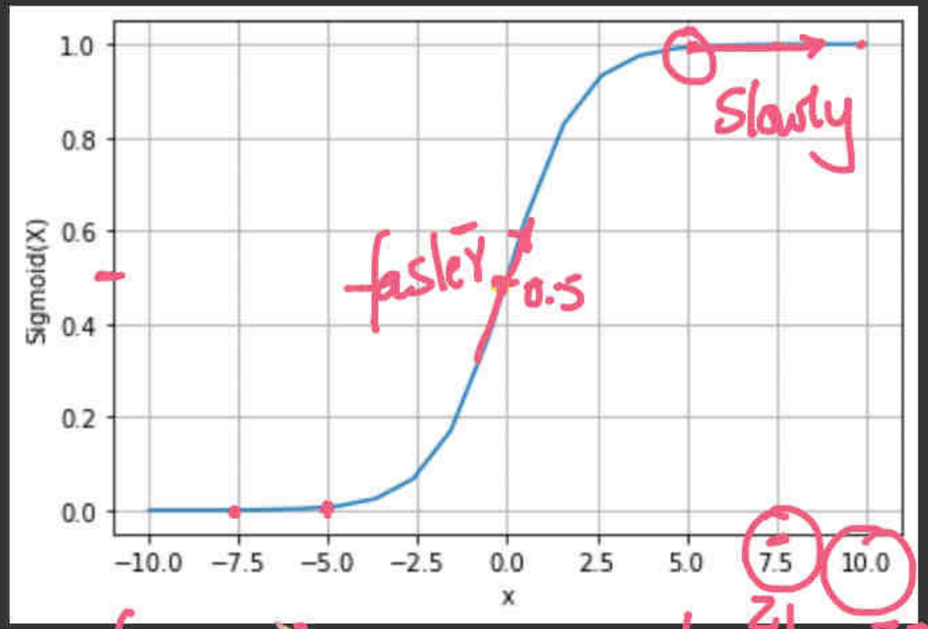
colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=DI6u91buPqyH

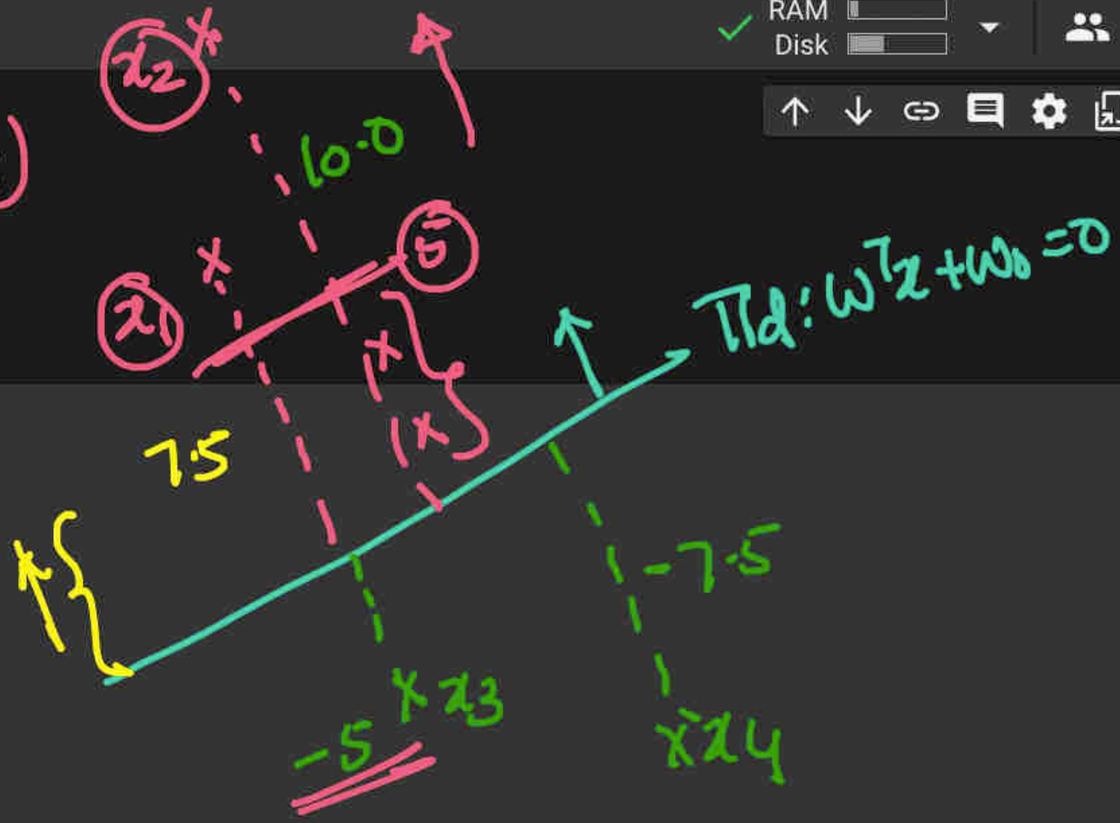
+ Code + Text

RAM Disk

```
x = np.linspace(-10, 10, 20)
z = sigmoid(x)
plt.plot(x, z)
plt.xlabel("x")
plt.ylabel("Sigmoid(X)")
plt.grid()
plt.show()
```

$\sigma(z_2) > \sigma(z_1)$





z_i : signed dist b/w Πd & x_i

```
[ ] # log loss
    y= 1
    yhat=0.9
```

Context:

$\mathcal{D} = \{(x_i, y_i)_{i=1}^{n}; \quad y_i \in \{0,1\}; \quad x_i \in \mathbb{R}^d\}$

Logistic Regr

① Linear Seperator (function) $\quad \pi_d: \quad w^T x + w_0 = 0$

② $w^T x_i + w_0 = z_i \in (-\infty, \infty) \xrightarrow{\text{squashing}} (0,1)$

$\sigma(z_i)$

$\hat{y}_i$

prob

③ $\begin{cases} y_i : 0 \text{ or } 1 \\ \hat{y}_i \in (0,1) \end{cases}$

Compare y_i vs $\hat{y}_i$

$\boxed{\text{Log-loss}}$

Chrome File Edit View History Bookmarks Profiles Tab Window Help 21:40

Linear Reg:

$$\mathcal{D} = \{(x_i, y_i)_{i=1}^{n} \; \ldots \; y_i \in \mathbb{R}\}$$

① Lr. fn: $\overbrace{w^T x_i + w_0}^{} \in (-\infty, \infty)$ No-squashing needed

$$w^T x_i + w_0 = \hat{y}_i$$

② y_i vs $\hat{y}_i$

$$\boxed{\text{Squared loss}} \quad \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

12 / 14

Chrome File Edit View History Bookmarks Profiles Tab Window Help 21:47

Log-loss!

Compare

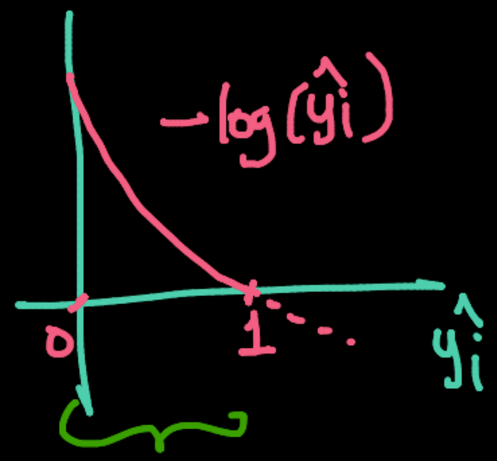
observed

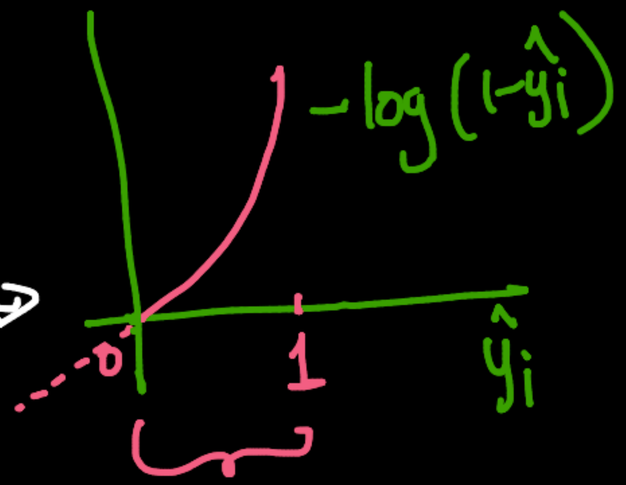
y_i : 0 or 1 $\leftarrow x_i$

$\hat{y}_i \in (0,1)$

predicted

$$\text{log-loss} = \begin{cases} -\log(\hat{y}_i) & \text{if } y_i = 1 \\ \boxed{-\log(1-\hat{y}_i)} & \text{if } y_i = 0 \end{cases}$$





13 / 17

Chrome File Edit View History Bookmarks Profiles Tab Window Help 21:44

Logistic Regression.ipynb - Co | plot(-log(x)) - Google Search | plot(-log(1-x)) - Google Searc

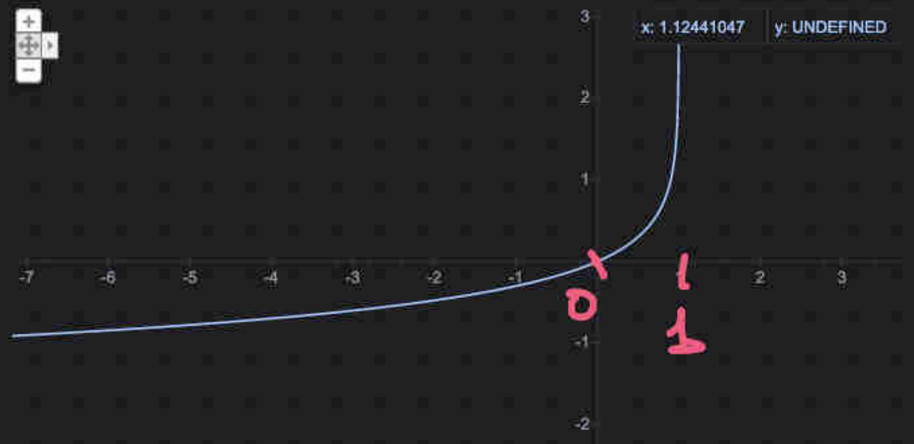
google.com/search?q=plot(-log(1-x))&rlz=1C5CHFA_enIN958IN958&oq=plot(-log(1-x))&aqs=chrome..69i57j0i30j0i390.11655j0j7&sourceid=chrome&ie=UTF-8

Google plot(-log(1-x))

All Images Videos Maps News More Tools

About 1,50,00,00,000 results (0.58 seconds)

Graph for -log(1-x)



More info

https://www.mathway.com › popular-problems › Algebra

Graph y = log of 1-x | Mathway

The **log** function can be graphed using the vertical asymptote at **x=1** **x = 1** and the points (0,0), (−1,0.30102999),(−2,0.47712125) (0 , 0) , (- 1 ...

https://www.geogebra.org › kbSBpWEB

log (x) vs. log (1/x) - GeoGebra

log (**x**) vs. **log** (**1/x**). Author: mcdull. Topic: Logarithmic Functions. **log** (**x**) vs. **log** (**1/x**). GeoGebra Applet Press Enter to start activity ...

https://www.wolframalpha.com › input › i=plot+log(1-x)

plot log(1-x) - WolframAlpha

$\hat{y}_i$

$y_i = 0$

$\hat{y}_i = 0$

$-\log(1-\hat{y}_i) \to 0 = \text{log-loss}$

$\begin{cases} y_i = 0 \\ \hat{y}_i = 1 \end{cases}$

log-loss : v. high

14 / 16

Chrome File Edit View History Bookmarks Profiles Tab Window Help 21:48

why $\boxed{\text{log-loss}}$ → logistic regr ← probability (MLE)

squared loss$_i$: $(y_i - \hat{y}_i)^2$ vs $(y_i - \hat{y}_i)^4$

probability estalo (MLE)

Chrome File Edit View History Bookmarks Profiles Tab Window Help

18 / 18

Chrome File Edit View History Bookmarks Profiles Tab Window Help

18 / 18

Chrome File Edit View History Bookmarks Profiles Tab Window Help 21:57

$$\text{log-loss} = \begin{cases} -\log(\hat{y}_i) & \text{if } y_i = 1 \\ -\log(1-\hat{y}_i) & \text{if } y_i = 0 \end{cases}$$

hack

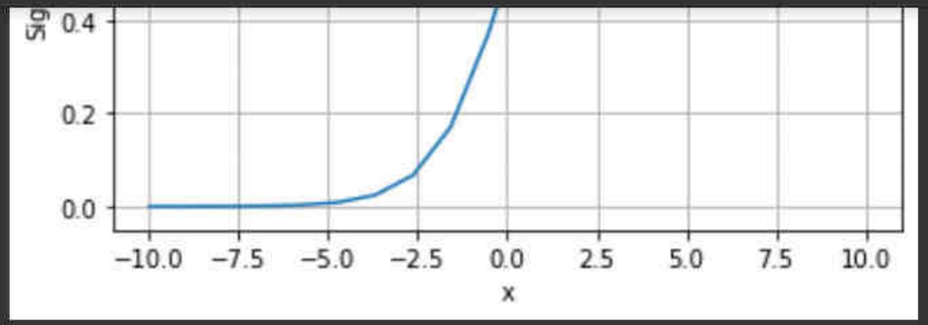
$$= -\overset{1}{y_i} \log(\hat{y}_i) - (1-y_i) \log(1-\hat{y}_i)$$

$$\begin{cases} y_i = 1 \Rightarrow -\log(\hat{y}_i) \\ y_i = 0 \Rightarrow 0 - 1 \log(1-\hat{y}_i) \end{cases}$$

18 / 18

Chrome File Edit View History Bookmarks Profiles Tab Window Help 21:58
Logistic Regression.ipynb - Co | plot(-log(x)) - Google Search | plot(-log(1-x)) - Google Searc
colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=Lk_DJSO3Bcbr
Update
+ Code + Text
RAM
Disk

[2]




```
# log loss
y = 1
yhat=0.9

print(-y*math.log(yhat)-(1-y)*math.log(1-yhat))
```

$$\begin{cases} -y \log(\hat{y}) \\ -(1-y)\log(1-\hat{y}) \end{cases}$$

```
0.10536051565782628
```

```
y = 1
yhat=0.99

print(-y*math.log(yhat)-(1-y)*math.log(1-yhat))
```

```
0.01005033585350145
```

```
y = 1
```

19 / 19

$$\mathcal{D}_{Tr} = \{(x_i, y_i)_{i=1}^{n} \; \ldots \}$$

Logistic Reg: (Classfn)

$$\min_{w_j} \sum_{i=1}^{n} \left[-y_i \log(\hat{y}_i) - \underbrace{(1-y_i)\log(1-\hat{y}_i)}_{\text{log-loss}_i} \right]$$

Linear Reg: (reg)

$$\min_{w_j} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Logistic regression → classification

→ x_i → real-value (0, 1) prob

$p(y_i = 1 | x_i)$

$$\mathcal{D}_{Tr} = \{(x_i, y_i)_{i=1}^{n} \dots\}$$

$$\min_{\underline{w_j}} \Big[\overbrace{\sum_{i=1}^{n} \Big[-y_i \log(\hat{y}_i) - (1-y_i) \log(1-\hat{y}_i) \Big]}^{\mathcal{L}} \Big] + \lambda \sum_j w_j^2$$

$\hat{y}_i \to \sigma(\underline{w}^T x_i + \underline{w}_0)$

↳ L_2-reg : $w^T w = \|w\|^2$

no non-linear features

↳ do I need reg? → Yes

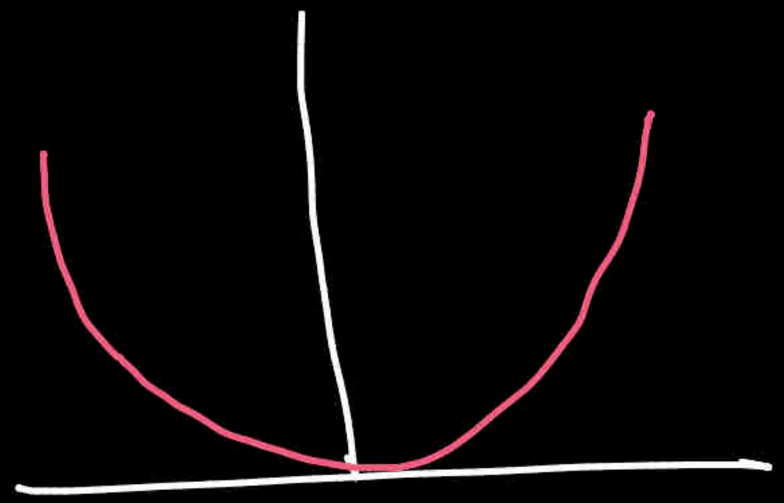
↳ [why?] do NOT overfit to noise

Optimization $\longrightarrow$ Gradient Descent

$$\frac{\partial \mathcal{L}}{\partial w_j} \quad \&$$

$$\sigma(z_i) = \frac{1}{1+\exp(-z_i)}$$

$$(y_i - \hat{y}_i)^2$$



$$\overbrace{y_i - \hat{y}_i}^{} = e_i$$

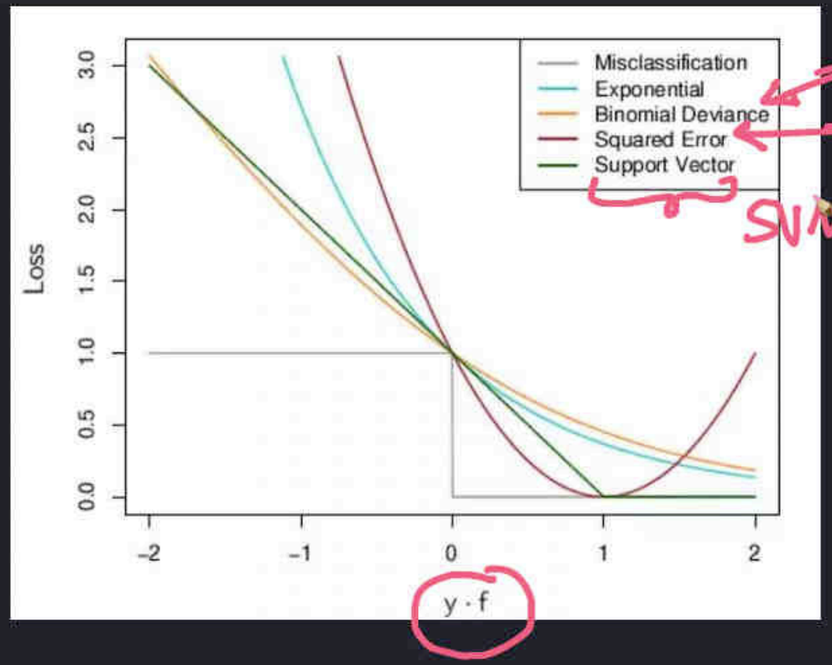
log-loss:

(next-class)

[?]

$(y_i \cdot z_i)$

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:12
Logistic Regression.ipynb - Co | loss functions ml log loss vs s | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc
yuan-du.com/post/2020-12-13-loss-functions/decision-theory/ Update



Loss Functions: "Loss vs y*f(x); $y = \pm 1$, the prediction is f, with class prediction sign(f)"

1. Hinge loss

The hinge loss is a loss function used for "maximum-margin" classification, most notably for support vector machine (SVM).It's equivalent to minimize the loss function $L(y, f) = [1 - yf]_+$.

With $f(x) = h(x)^T\beta + \beta_0$, the optimization problem is loss + penalty:

25 / 25

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:15

Q

exercise

$\sigma(z)$: chain rule & algebra

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z)\,(1-\sigma(z))$$

$$\frac{1}{n}\sum_{i=1}^{n}\overbrace{(y_i-\hat{y}_i)^2}$$ → convex (DO NOT WORRY)

$$\sum_{i=1}^{n}\overbrace{\text{log-loss}_i}$$

Prob. of logistic reg (MLE)

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:18

$$\mathcal{D} = \{(x_i, y_i)_{i=1}^{n}\} \qquad y_i \in (0,1)$$

1. Lr. sep/fn: $w^T x_i + w_0 = z_i$: $\Pi\partial$

2. $z_i \in (-\infty, \infty) \xrightarrow[\sigma(z_i) = \hat{y}_i]{\text{squash}} (0,1)$: prob $p(y_i = 1 \mid x_i)$

3. y_i vs $\hat{y}_i$: log-loss:

28 / 28

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:20

4 opt: $\min_{w_j} \sum_{i=1}^{n} \text{log loss}_i + \lambda \sum_j w_j^2$

params ← w_j's → linear sep / function

5 hyperparam: overfitting & underfitting

λ

plots : best λ

29 / 29

Chrome File Edit View History Bookmarks Profiles Tab Window Help

22:20

30 / 30

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:21

6 $\lambda_{best} \rightarrow Model_{best}$

$D_{test} \rightarrow Model_{best} \rightarrow$ accuracy

$\hat{y}_i \in (0,1)$

default

$\hat{y}_i > 0.5 \rightarrow y_i = 1$

Thr

o/w $y_i = 0$

$y_i = 1$

$\hat{y}_i > 0.5$

$\downarrow$

1

✓

31 / 32

Test

100

accuracy

$y_{pred} = 1$ if $\hat{y}_i > 0.5$ else 0

y_i	$\hat{y}_i$	y_{pred}	
1	Match	1	→ ✓
0	Match	0	→ ✓
1		0	→ X
0		1	→ X

$$\frac{95 \text{ match}}{100} = 95\%$$

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic reg

$$\hat{y}_i = \sigma(w^T x_i + w_0)$$

log-loss

L_2-reg

lr. reg

$$\hat{y}_i = w^T x_i + w_0$$

sq. loss

L_2-reg

Logistic Regr → Math interpretations

odds

log loss & SQ-loss

[next-class]

Code = hands-on

34 / 34

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=mbBNqrRwBz4I

+ Code + Text

RAM Disk

Cust → features

Churn

1

[3]
```
# Churn prediction in telecom.
import numpy as np
import matplotlib.pyplot as plt
```

[4]
```
#https://drive.google.com/file/d/1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_/view?usp=sharing

id = "1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_"
print("https://drive.google.com/uc?export=download&id=" + id)
```

```
https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_
```

[5]
```
!wget "https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_" -O Churn.csv
```

```
--2022-04-27 13:58:15--  https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_
Resolving drive.google.com (drive.google.com)... 74.125.31.100, 74.125.31.139, 74.125.31.138, ...
Connecting to drive.google.com (drive.google.com)|74.125.31.100|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-0g-ag-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1/q25kkf7sii9poi
Warning: wildcards not supported in HTTP.
--2022-04-27 13:58:15--  https://doc-0g-ag-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1
Resolving doc-0g-ag-docs.googleusercontent.com (doc-0g-ag-docs.googleusercontent.com)... 142.250.98.132, 2607:f8b0:4
Connecting to doc-0g-ag-docs.googleusercontent.com (doc-0g-ag-docs.googleusercontent.com)|142.250.98.132|:443... con
HTTP request sent, awaiting response... 200 OK
```

35 / 35

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=mbBNqrRwBz4I

+ Code + Text

RAM Disk

Churn → 1, → 0

[3]
```
# Churn prediction in telecom.
import numpy as np
import matplotlib.pyplot as plt
```

[4]
```
#https://drive.google.com/file/d/1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_/view?usp=sharing

id = "1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_"
print("https://drive.google.com/uc?export=download&id=" + id)
```

```
https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_
```

[5]
```
!wget "https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_" -O Churn.csv
```

```
--2022-04-27 13:58:15--  https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_
Resolving drive.google.com (drive.google.com)... 74.125.31.100, 74.125.31.139, 74.125.31.138, ...
Connecting to drive.google.com (drive.google.com)|74.125.31.100|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-0g-ag-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1/q25kkf7sii9poi
Warning: wildcards not supported in HTTP.
--2022-04-27 13:58:15--  https://doc-0g-ag-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1
Resolving doc-0g-ag-docs.googleusercontent.com (doc-0g-ag-docs.googleusercontent.com)... 142.250.98.132, 2607:f8b0:4
Connecting to doc-0g-ag-docs.googleusercontent.com (doc-0g-ag-docs.googleusercontent.com)|142.250.98.132|:443... con
HTTP request sent, awaiting response... 200 OK
```

36 / 36

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=-Emm5e89Ubji

+ Code + Text

RAM Disk

```
churn["Churn"].value_counts()
```

3333

```
0    2850
1     483
Name: Churn, dtype: int64
```

→ not-churn

```
[21] churn.columns
```

```
Index(['Account Length', 'VMail Message', 'Day Mins', 'Eve Mins', 'Night Mins',
       'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
       'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
       'Night Charge', 'Intl Calls', 'Intl Charge', 'State', 'Area Code',
       'Phone'],
      dtype='object')
```

```
[9] # Basic EDA
    import seaborn as sns

    sns.boxplot(x='Churn', y='Day Mins', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b7e267d0>
```

37 / 37

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:29

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=-Emm5e89Ubji

+ Code + Text

RAM Disk

```
churn["Churn"].value_counts()
```

```
0    2850
1     483
Name: Churn, dtype: int64
```

```
[21] churn.columns
```

```
Index(['Account Length', 'VMail Message', 'Day Mins', 'Eve Mins', 'Night Mins',
       'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
       'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
       'Night Charge', 'Intl Calls', 'Intl Charge', 'State', 'Area Code',
       'Phone'],
      dtype='object')
```

```
[9] # Basic EDA
    import seaborn as sns

    sns.boxplot(x='Churn', y='Day Mins', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b7e267d0>
```

38 / 38

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:29

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=-Emm5e89Ubji

+ Code + Text

RAM Disk

```
churn["Churn"].value_counts()
```

```
0    2850
1     483
Name: Churn, dtype: int64
```

EDA

Day Mins Churn

```
[21] churn.columns
```

```
Index(['Account Length', 'VMail Message', 'Day Mins', 'Eve Mins', 'Night Mins',
       'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
       'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
       'Night Charge', 'Intl Calls', 'Intl Charge', 'State', 'Area Code',
       'Phone'],
      dtype='object')
```

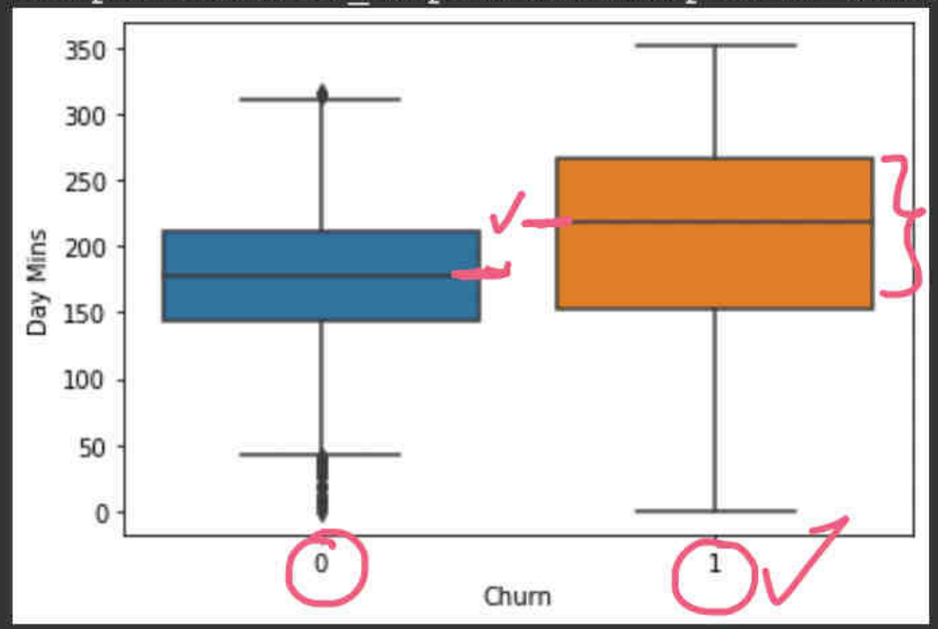
```
[9] # Basic EDA
    import seaborn as sns

    sns.boxplot(x='Churn', y='Day Mins', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b7e267d0>
```

39 / 39

```
# Basic EDA, not comprehensive
import seaborn as sns

sns.boxplot(x='Churn', y='Day Mins', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b7e267d0>
```



Day Mins

```
[10] # simple correlation, not full collienarity

sns.pairplot(data=churn, y_vars=["Day Mins"], x_vars=['Account Length', 'VMail Message', 'Eve Mins', 'Night Mins',
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:32

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=XxVRKCWMUgIB

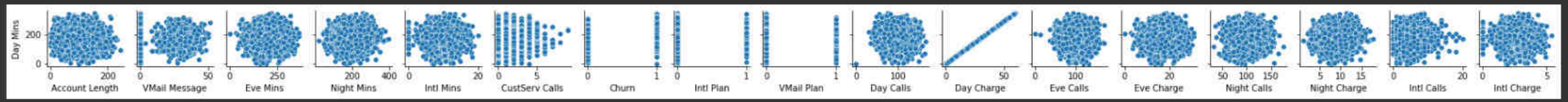
+ Code + Text

RAM Disk

day_mins

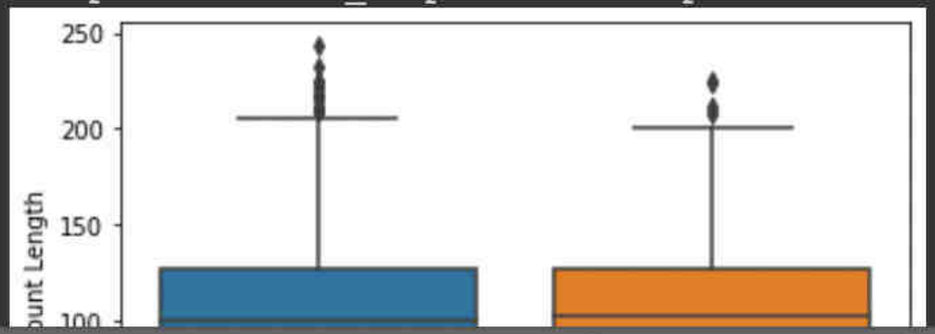
```
[10] # simple correlation, not full collienarity

     sns.pairplot(data=churn, y_vars=["Day Mins"], x_vars=['Account Length', 'VMail Message', 'Eve Mins', 'Night Mins',
            'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
            'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
            'Night Charge', 'Intl Calls', 'Intl Charge'], height=1.5, aspect=1)
     plt.show()
     # Day charge vs Day Mins
```

not VIF




```
[11] sns.boxplot(x = 'Churn', y= 'Account Length', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b5115350>
```



41 / 41

Chrome File Edit View History Bookmarks Profiles Tab Window Help

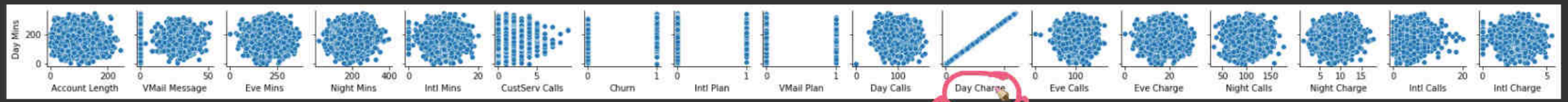
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=XxVRKCWMUgIB

+ Code + Text

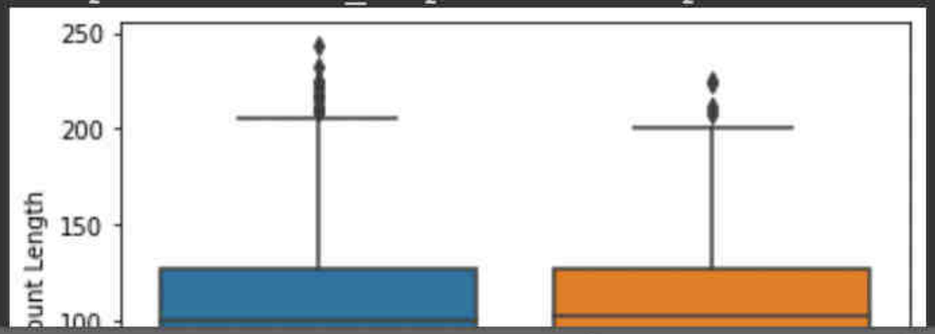
RAM
Disk

[10]
```
# simple correlation, not full collienarity

sns.pairplot(data=churn, y_vars=["Day Mins"], x_vars=['Account Length', 'VMail Message', 'Eve Mins', 'Night Mins',
        'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
        'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
        'Night Charge', 'Intl Calls', 'Intl Charge'], height=1.5, aspect=1)
plt.show()
# Day charge vs Day Mins
```



[11]
```
sns.boxplot(x = 'Churn', y= 'Account Length', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b5115350>
```



42 / 42

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc

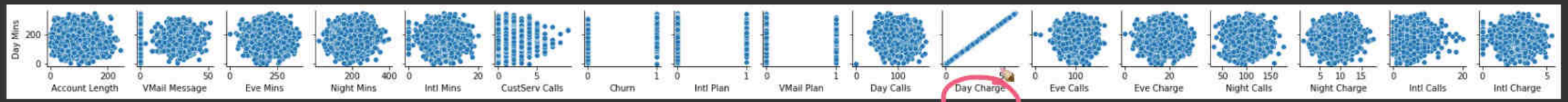
colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=XxVRKCWMUgIB

+ Code + Text

RAM
Disk

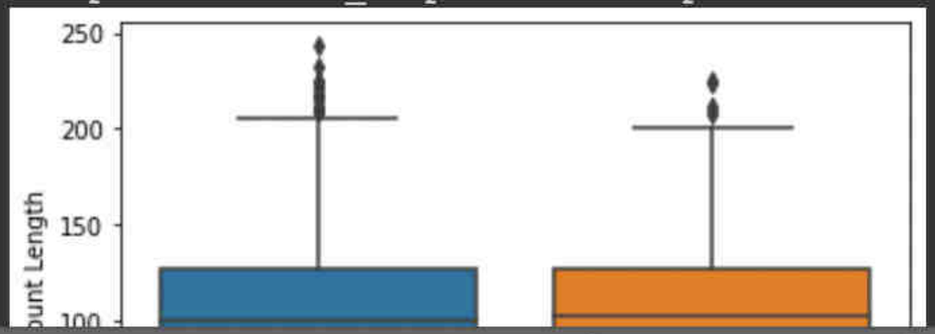
$w_i f_i$ $w_j f_j$

```
[10] # simple correlation, not full collienarity

     sns.pairplot(data=churn, y_vars=["Day Mins"], x_vars=['Account Length', 'VMail Message', 'Eve Mins', 'Night Mins',
            'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
            'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
            'Night Charge', 'Intl Calls', 'Intl Charge'], height=1.5, aspect=1)
     plt.show()
     # Day charge vs Day Mins
```



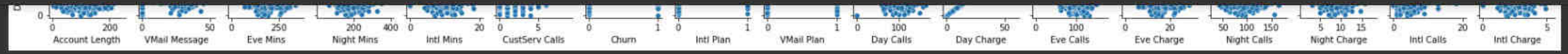

```
[11] sns.boxplot(x = 'Churn', y= 'Account Length', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b5115350>
```



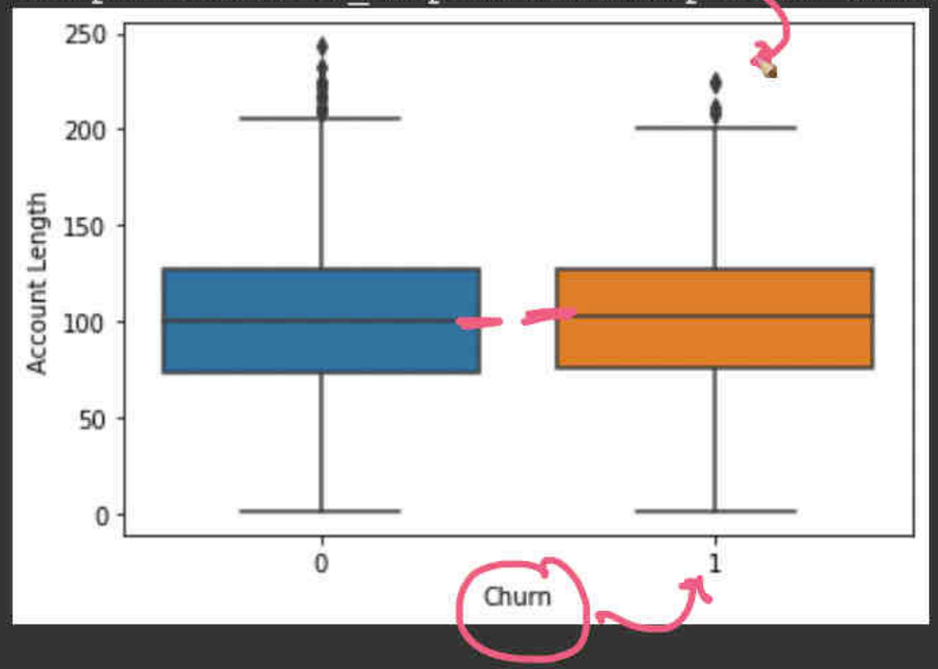
43 / 43

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:35
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc
colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=XxVRKCWMUgIB
+ Code + Text
RAM
Disk




```
[11] sns.boxplot(x = 'Churn', y= 'Account Length', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b5115350>
```

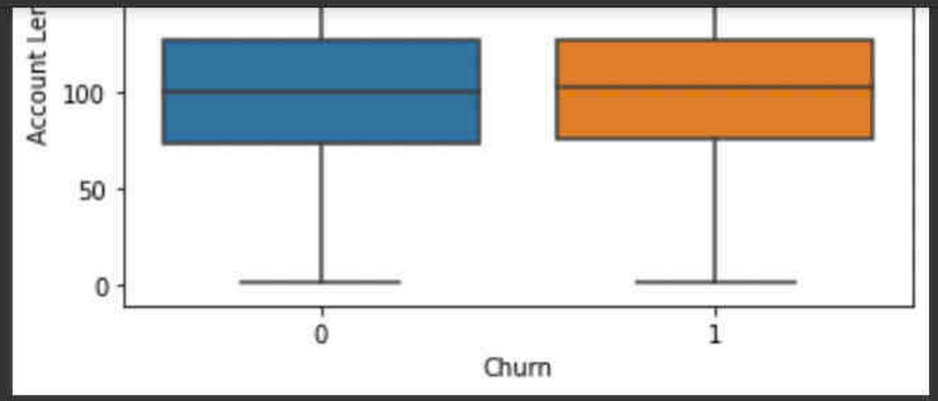



```
[12] # Skipping rest of the EDA :
     # Exercsie for students complete the rest of the EDA to find out which variables has some relation with target vari
```

44 / 44

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:36
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc
colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=f-kVXG8cVIkF
+ Code + Text
RAM
Disk

[11]



remove useless features
remove highly correlated :..

```
# Skipping rest of the EDA :
# Exercise for students complete the rest of the EDA to find out which variables has some relation with target vari
```

```
[22] # using a few features as an example. We can drop all useless features.
     cols = ['Day Mins', 'Eve Mins', 'CustServ Calls', 'Intl Plan','VMail Message']
```

```
[30] y=churn["Churn"]
```

```
[31] X = churn[cols]
```

```
[32] X.shape
```

(3333, 5)

45 / 45

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=aArfBDD4Wpx0

+ Code + Text

RAM Disk

[32]
```
X.shape
```

```
(3333, 5)
```

80% — 60% Tr, 20% CV

20% — Test

3333

```
# Train, CV, test split
from sklearn.model_selection import train_test_split
#0.6, 0.2, 0.2 split

X_tr_cv, X_test, y_tr_cv, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

X_train, X_val, y_train, y_val = train_test_split(X_tr_cv, y_tr_cv, test_size=0.25, random_state=42) # 0.25 x 0.8 =
```

[69]
```
X_train.shape
```

```
(1999, 5)
```

[70]
```
#scaling the data
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
```

```
StandardScaler()
```

46 / 46

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=TSkF-YsCcVRa

+ Code + Text

RAM Disk

[68]
```
X_tr_cv, X_test, y_tr_cv, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

X_train, X_val, y_train, y_val = train_test_split(X_tr_cv, y_tr_cv, test_size=0.25, random_state=42) # 0.25 x 0.8 =
```

[69]
```
X_train.shape
```

```
(1999, 5)
```

m_j, s_j from Train

```
#scaling the data
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
```

```
StandardScaler()
```

[71]
```
from sklearn.linear_model import LogisticRegression
#https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
LogisticRegression()
```

[72]
```
model.coef_
```

47 / 47

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:40
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR
scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
Update

Prev Up Next

scikit-learn 1.0.2
Other versions

Please cite us if you use the software.

sklearn.linear_model.LogisticRegression
Examples using sklearn.linear_model.LogisticRe

reg

sklearn.linear_model.LogisticRegression

class sklearn.linear_model.LogisticRegression(*penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None*) [source]

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

Read more in the User Guide.

Parameters: **penalty : *{'l1', 'l2', 'elasticnet', 'none'}, default='l2'***

Specify the norm of the penalty:

- `'none'`: no penalty is added;
- `'l2'`: add a L2 penalty term and it is the default choice;
- `'l1'`: add a L1 penalty term;
- `'elasticnet'`: both L1 and L2 penalty terms are added.

48 / 48

he parameter `solver` below, to know

Toggle Menu

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:42
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR
scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
Update

Specify the norm of the penalty:

- `'none'`: no penalty is added;
- `'l2'`: add a L2 penalty term and it is the default choice;
- `'l1'`: add a L1 penalty term;
- `'elasticnet'`: both L1 and L2 penalty terms are added.

$C = 1/\lambda$ $\lambda = \frac{1}{C}$

> **Warning:** Some penalties may not work with some solvers. See the parameter `solver` below, to know the compatibility between the penalty and solver.

New in version 0.19: l1 penalty with SAGA solver (allowing 'multinomial' + L1)

dual : *bool, default=False*
Dual or primal formulation. Dual formulation is only implemented for l2 penalty with liblinear solver. Prefer dual=False when n_samples > n_features.

$$\sum_{i=1}^{n} \log loss_i + \frac{1}{C} \sum_j w_j^2$$

tol : *float, default=1e-4*
Tolerance for stopping criteria.

C : *float, default=1.0*
Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.

fit_intercept : *bool, default=True*
Specifies if a constant (a.k.a. bias or intercept) should be added to the decision function.

intercept_scaling : *float, default=1*
Useful only when the solver 'liblinear' is used and self.fit_intercept is set to True. In this case, x becomes [x, self.intercept_scaling], i.e. a "synthetic" feature with constant value equal to intercept_scaling is appended to the instance vector. The intercept becomes `intercept_scaling * synthetic_feature_weight`.

Note! the synthetic feature weight is subject to l1/l2 regularization as all other features. To lessen the effect of regularization on synthetic feature weight (and therefore on the intercept) intercept_scaling has to be increased

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM Disk

[70]
```
#scaling the data
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
```

```
StandardScaler()
```

$\lambda = \frac{1}{C}$

l2-reg: $C = 1 \Rightarrow \boxed{\lambda = 1}$

```
from sklearn.linear_model import LogisticRegression
#https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
LogisticRegression()
```

[72]
```
model.coef_
```

```
array([[ 0.01171028,  0.00628069,  0.51870002,  1.97741128, -0.03885603]])
```

[73]
```
model.intercept_
```

```
array([-6.3286969])
```

50 / 50

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM
Disk

[70]
```
#scaling the data
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
```

```
StandardScaler()
```

```
from sklearn.linear_model import LogisticRegression
#https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
LogisticRegression()
```

highest

[72]
```
model.coef_
```

```
array([[ 0.01171028,  0.00628069,  0.51870002,  1.97741128, -0.03885603]])
```

w_1 w_2 w_3 w_4 w_5

[73]
```
model.intercept_
```

```
array([-6.3286969])
```

w_0

51 / 51

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM Disk

[87]
```
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

lambda

$C = \frac{1}{\lambda}$

[88]
```
len(val_scores)
```

```
500
```

[90]
```
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
```

52 / 52

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:46

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM Disk

[87]
```
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

0.01 0.11 .. .. 50.0

[88]
```
len(val_scores)
```

```
500
```

[90]
```
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
```

53 / 53

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:46

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM Disk

[87]
```
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

[88]
```
len(val_scores)
```

```
500
```

[90]
```
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
```

54 / 54

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM Disk

[87]
```
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

[88]
```
len(val_scores)
```

```
500
```

[90]
```
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
```

55 / 55

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM Disk

[87]
```
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
    scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
    scaled_lr.fit(X_train, y_train)
    train_score = scaled_lr.score(X_train, y_train)
    val_score = scaled_lr.score(X_val, y_val)
    train_scores.append(train_score)
    val_scores.append(val_score)
```

metrics

score: accuracy

[88]
```
len(val_scores)
```

```
500
```

[90]
```
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
```

56 / 56

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=zmREN5hSW-yX

+ Code + Text

RAM
Disk

[87]
```
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

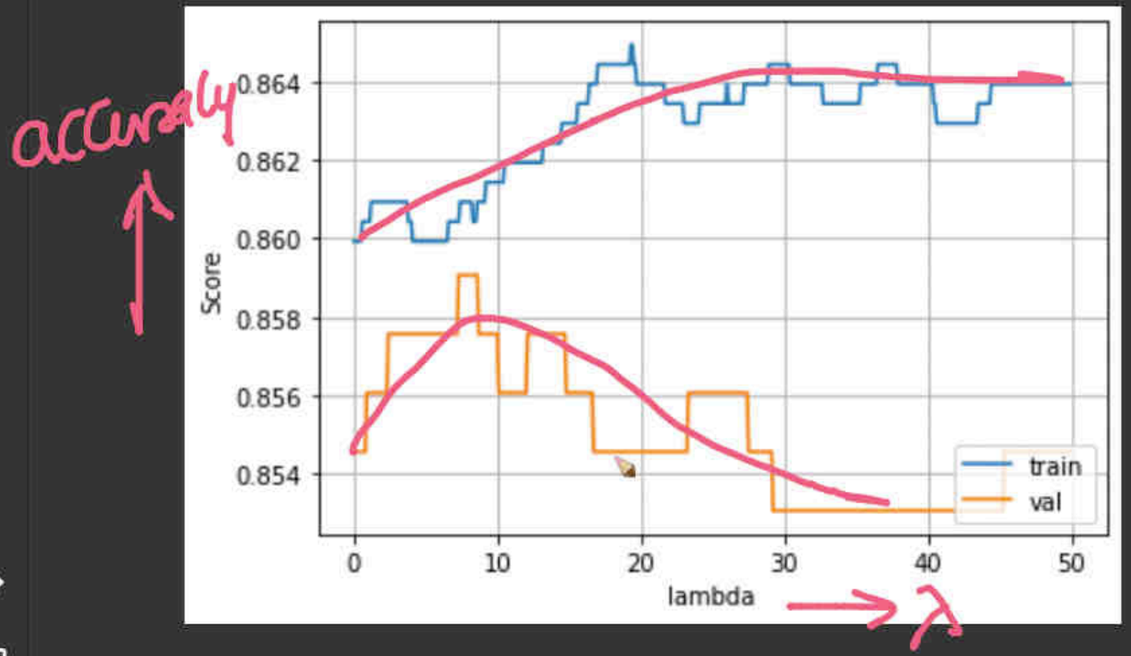
[88]
```
len(val_scores)
```

```
500
```

[90]
```
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
```

57 / 57

```
[90] plt.figure()
     plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
     plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
     plt.legend(loc='lower right')
     plt.xlabel("lambda")
     plt.ylabel("Score")
     plt.grid()
     plt.show()
```




```
[91] np.argmax(val_scores)
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=8IsMQRHxe9_f

+ Code + Text

RAM Disk

```
[102] val_scores[73]
```

```
0.8590704647676162
```

Tr → w_j

```
[103] l_best = 0.01*73*0.1
```

Val → λ_{best} → $Model_{best}$

```
[104] # Model with lambda=
      scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/l_best))
      scaled_lr.fit(X_train, y_train)
```

```
Pipeline(steps=[('standardscaler', StandardScaler()),
                ('logisticregression',
                 LogisticRegression(C=13.698630136986303))])
```

Test → $Model_{best}$ →

```
test_score = scaled_lr.score(X_test, y_test)
print(test_score)
```

```
0.8515742128935532
```

59 / 59

```
array([-6.3286969])
```

```python
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

μ_j σ_j

X train → scaler → param Logistic Reg ($C=1/\lambda$) → score

X Val

[88]
```python
len(val_scores)
```

```
500
```

[90]
```python
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
```

```
array([-6.3286969])
```

```python
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 50.0, 0.1):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

[88]
```python
len(val_scores)
```

```
500
```

[90]
```python
plt.figure()
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
```

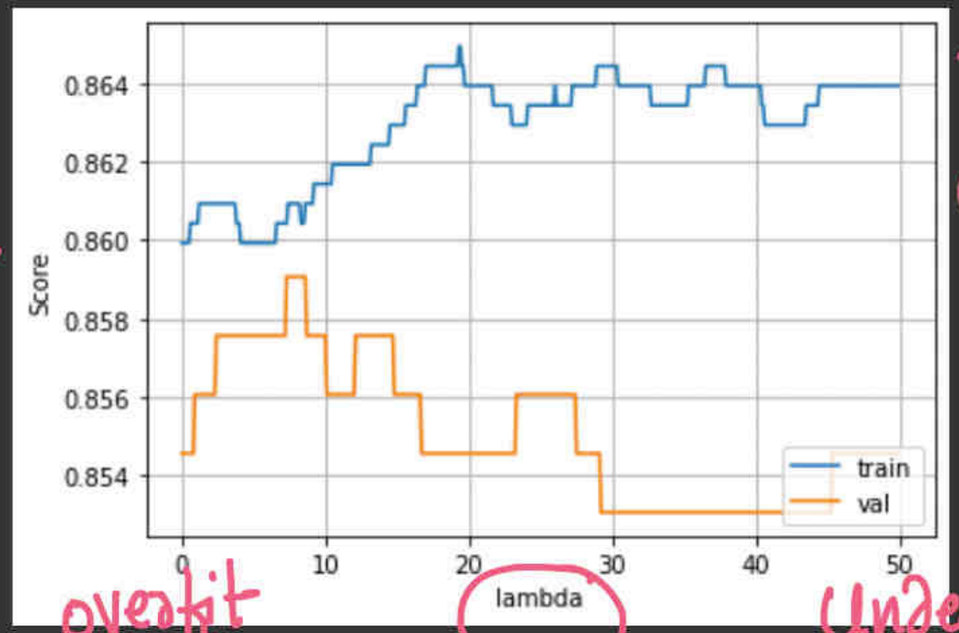
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=X2vZAFjWc-8E

+ Code + Text

RAM Disk

[90]
```
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
plt.ylabel("Score")
plt.grid()
plt.show()
```



acc

overfit

underfit

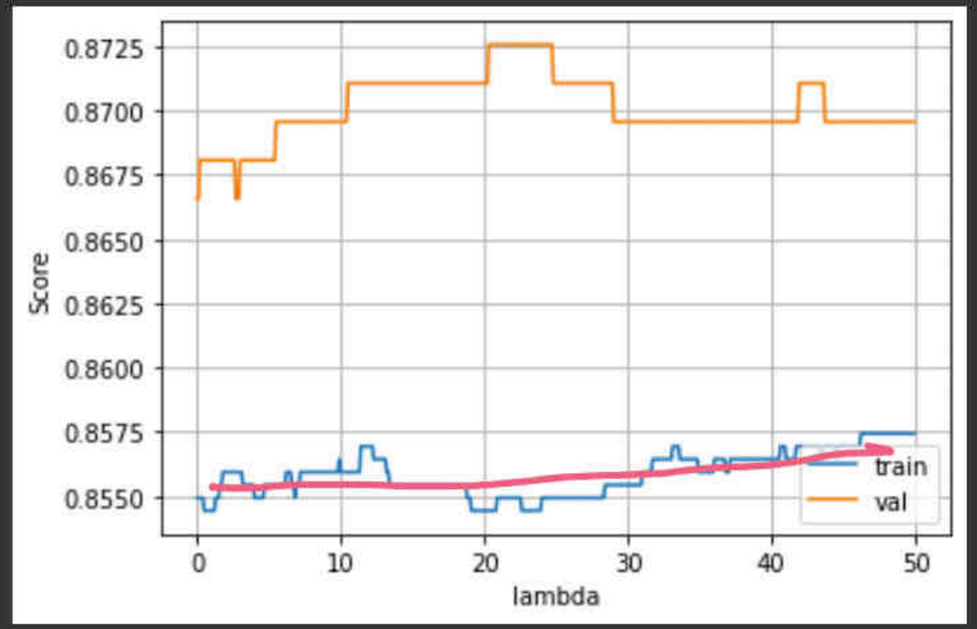
λ↑

[91]
```
np.argmax(val_scores)
```

62 / 62

```
plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
plt.ylabel("Score")
plt.grid()
plt.show()
```

random-state



60-20-20

λ

```
[91] np.argmax(val_scores)
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=X2vZAFjWc-8E

+ Code + Text

RAM Disk

```
array([-6.3286969])
```

```
# Hyper-pram tuning
from sklearn.pipeline import make_pipeline
train_scores = []
val_scores = []
scaler = StandardScaler()

for la in np.arange(0.01, 5000.0, 10):
  scaled_lr = make_pipeline( scaler, LogisticRegression(C=1/la))
  scaled_lr.fit(X_train, y_train)
  train_score = scaled_lr.score(X_train, y_train)
  val_score = scaled_lr.score(X_val, y_val)
  train_scores.append(train_score)
  val_scores.append(val_score)
```

λ: 0 ... ∞

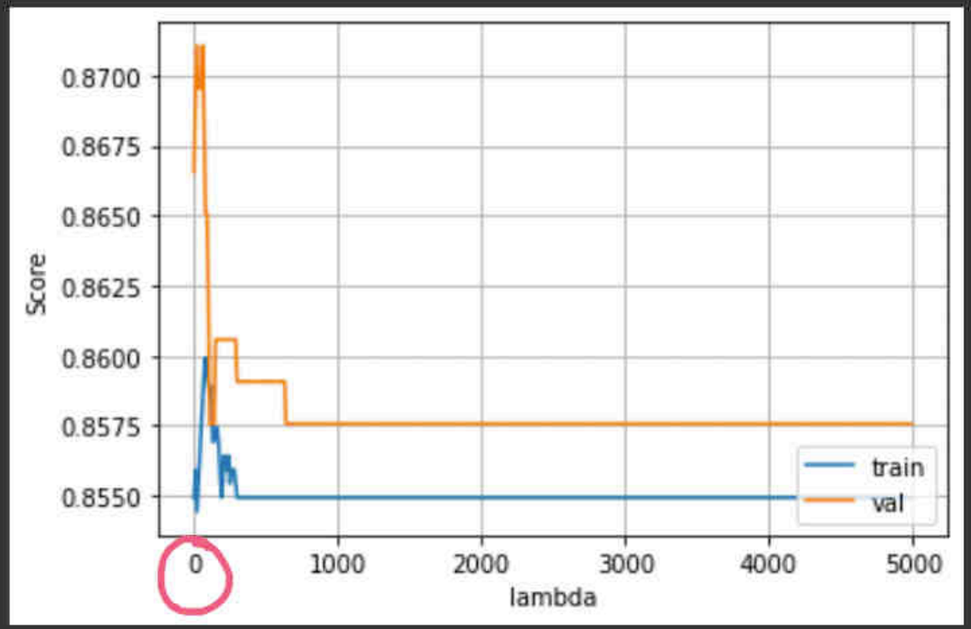
```
[88] len(val_scores)
```

```
500
```

```
[109] plt.figure()
      plt.plot(list(np.arange(0.01, 50.0, 0.1)), train_scores, label="train")
      plt.plot(list(np.arange(0.01, 50.0, 0.1)), val_scores, label="val")
```

64 / 64

```
plt.grid()
plt.show()
```



alc

```
[91] np.argmax(val_scores)
```

```
73
```

```
[102] val_scores[73]
```

```
0.8590704647676162
```

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4lWhB_PXhP-xJ#scrollTo=sN5cWIS4eKZn

+ Code + Text

RAM Disk

```
memory usage: 546.9+ KB
```

```
[20] churn["Churn"].value_counts()
```

```
0    2850
1     483
Name: Churn, dtype: int64
```

85% → class 0
15% → class 1
} imbalanced dataset

```
[21] churn.columns
```

```
Index(['Account Length', 'VMail Message', 'Day Mins', 'Eve Mins', 'Night Mins',
       'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
       'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
       'Night Charge', 'Intl Calls', 'Intl Charge', 'State', 'Area Code',
       'Phone'],
      dtype='object')
```

```
[9] # Basic EDA, not comprehensive
    import seaborn as sns

    sns.boxplot(x='Churn', y='Day Mins', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b7e267d0>
```

66 / 66

Chrome File Edit View History Bookmarks Profiles Tab Window Help 22:59
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR
colab.research.google.com/drive/1G_D39wa40kVahULehDS4lWhB_PXhP-xJ#scrollTo=sN5cWIS4eKZn
+ Code + Text
RAM
Disk

```
memory usage: 546.9+ KB
```

[20]
```
churn["Churn"].value_counts()
```

```
0    2850
1     483
Name: Churn, dtype: int64
```

[21]
```
churn.columns
```

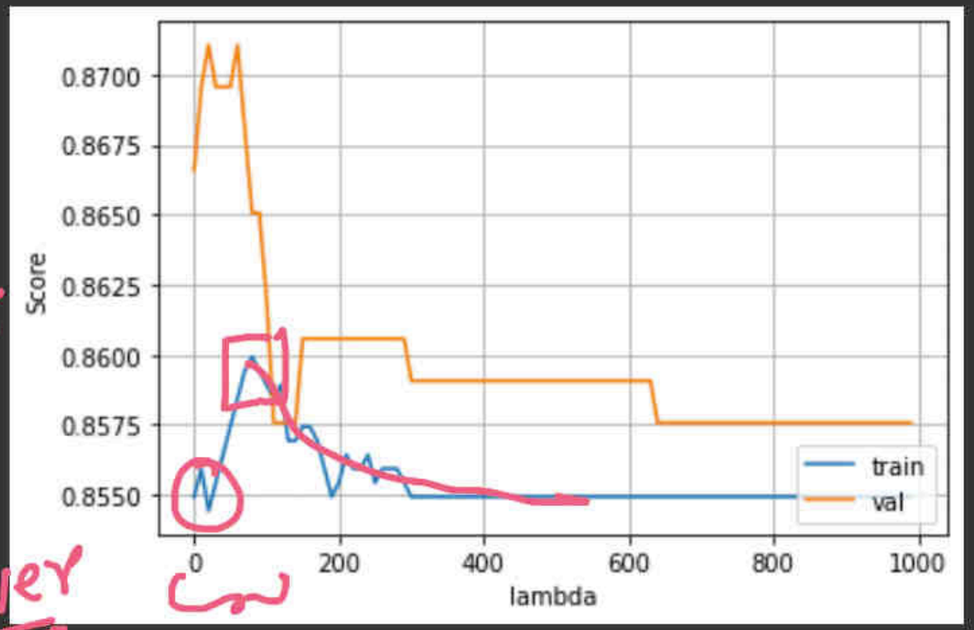
```
Index(['Account Length', 'VMail Message', 'Day Mins', 'Eve Mins', 'Night Mins',
       'Intl Mins', 'CustServ Calls', 'Churn', 'Intl Plan', 'VMail Plan',
       'Day Calls', 'Day Charge', 'Eve Calls', 'Eve Charge', 'Night Calls',
       'Night Charge', 'Intl Calls', 'Intl Charge', 'State', 'Area Code',
       'Phone'],
      dtype='object')
```

[9]
```
# Basic EDA, not comprehensive
import seaborn as sns

sns.boxplot(x='Churn', y='Day Mins', data = churn)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f21b7e267d0>
```

67 / 67

```
plt.plot(list(np.arange(0.01, 1000.0, 10)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
plt.ylabel("Score")
plt.grid()
plt.show()
```



acc

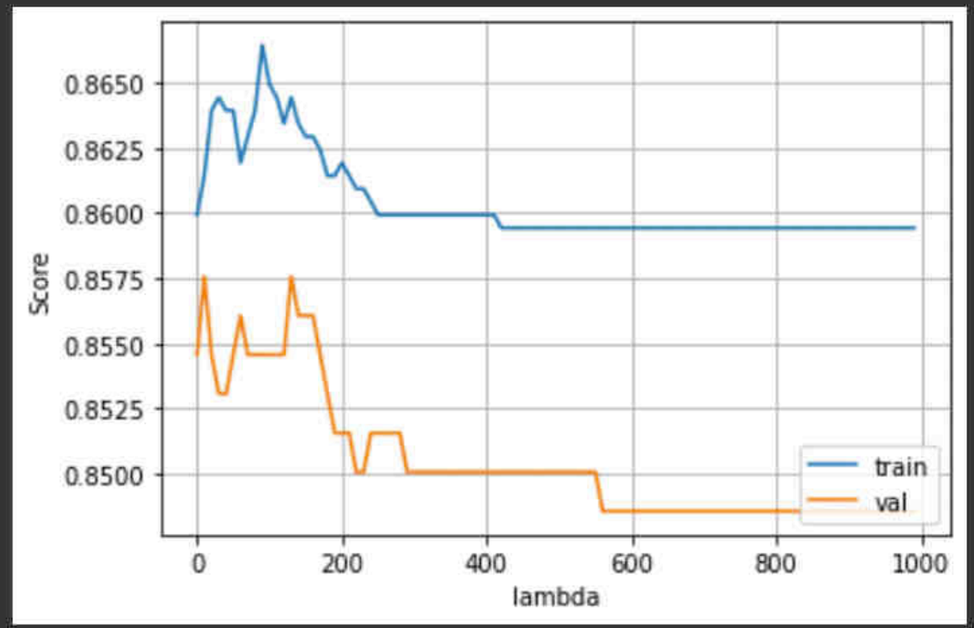
over

underfitting

```
[91] np.argmax(val_scores)
```

```
73
```

```
plt.plot(list(np.arange(0.01, 1000.0, 10)), train_scores, label="train")
plt.plot(list(np.arange(0.01, 1000.0, 10)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
plt.ylabel("Score")
plt.grid()
plt.show()
```



random-state

Tr - Te - CV

15%

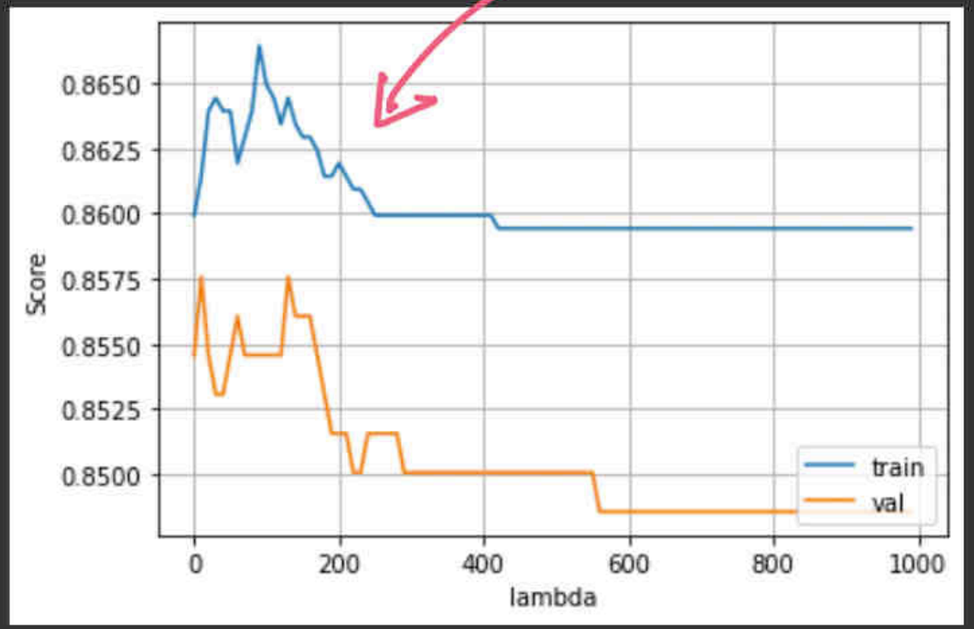
label = 1

```
[91] np.argmax(val_scores)
```

```
plt.grid()
plt.show()
```

[117]

random-split



15%

label=1 : 46 | 85%

Tr

Val → 96

Te

```
[91] np.argmax(val_scores)
```

```
73
```

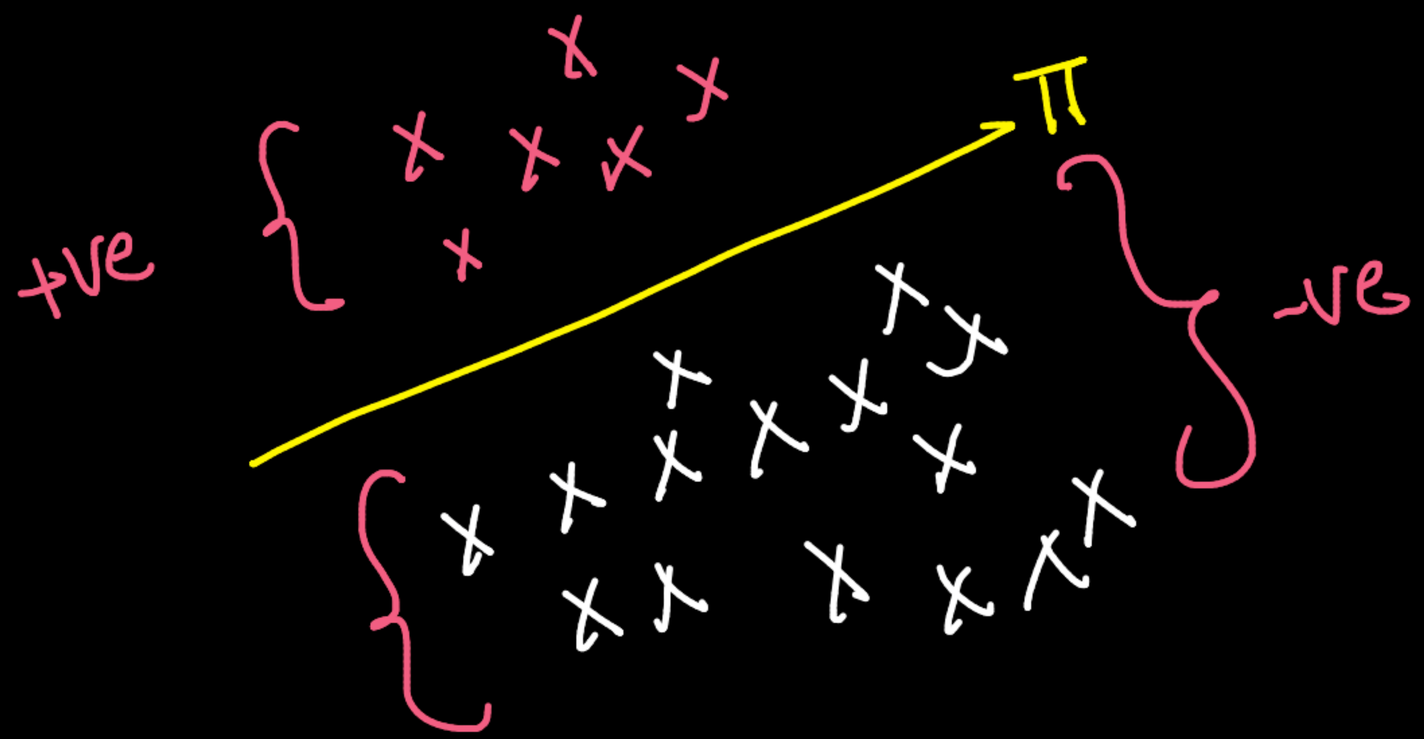
```
[102] val_scores[73]
```

```
0.8590704647676162
```

Train

$$\text{Churn} = y_i \begin{cases} \rightarrow 1 & (150) \\ \rightarrow 0 & (850) \end{cases}$$

$$\text{loss} = \overbrace{\sum_{i=1}^{n} \overbrace{\log\text{-loss}_i}}^{\min} + \text{reg}$$



Chrome File Edit View History Bookmarks Profiles Tab Window Help 23:11

imbalance

+ve

-ve

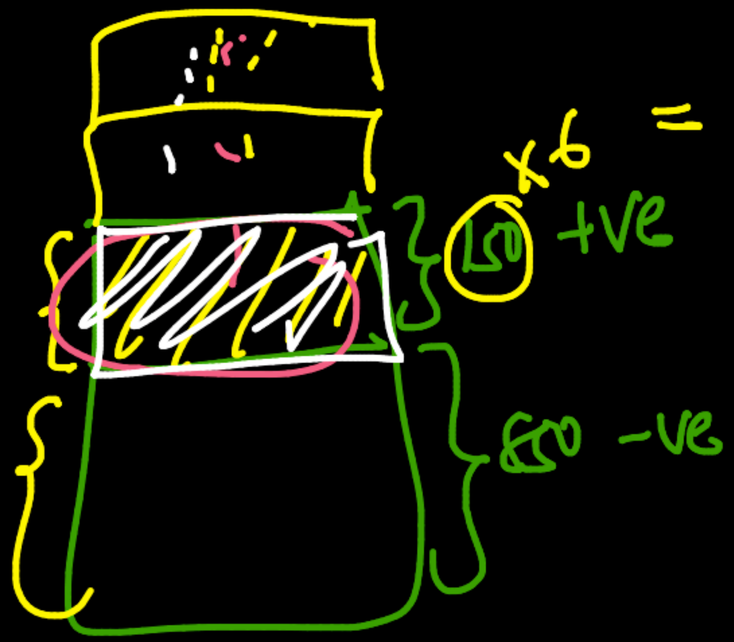
1

150 → 850

900

✓

✓ replicating



$$n = 850 + 900$$

$$\sum_{i=1} \text{log-loss}_i$$

72 / 75

Chrome File Edit View History Bookmarks Profiles Tab Window Help 23:11

2

$$\sum_{i=1}^{n} \log \cdot loss_i \; w_i$$

150 − 850

$$\frac{850}{150} \sim 6$$

$w_i = 6$ if $y_i = 1$

$w_i = 1$ if $y_i = 0$

Chrome File Edit View History Bookmarks Profiles Tab Window Help 23:12
Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Lea | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR
scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
Update

Prev Up Next

scikit-learn 1.0.2
Other versions

Please cite us if you use the software.

sklearn.linear_model.LogisticRegression
Examples using sklearn.linear_model.LogisticRe

sklearn.linear_model.LogisticRegression

class sklearn.linear_model.LogisticRegression(*penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None*) [source]

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

Read more in the User Guide.

Parameters:

penalty : *{'l1', 'l2', 'elasticnet', 'none'}, default='l2'*

Specify the norm of the penalty:

- `'none'`: no penalty is added;
- `'l2'`: add a L2 penalty term and it is the default choice;
- `'l1'`: add a L1 penalty term;
- `'elasticnet'`: both L1 and L2 penalty terms are added.

74 / 75

he parameter `solver` below, to know

Toggle Menu

Useful only when the solver 'liblinear' is used and self.fit_intercept is set to True. In this case, x becomes [x, self.intercept_scaling], i.e. a "synthetic" feature with constant value equal to intercept_scaling is appended to the instance vector. The intercept becomes `intercept_scaling * synthetic_feature_weight`.

Note! the synthetic feature weight is subject to l1/l2 regularization as all other features. To lessen the effect of regularization on synthetic feature weight (and therefore on the intercept) intercept_scaling has to be increased.

class_weight : *dict or 'balanced', default=None*

Weights associated with classes in the form `{class_label: weight}`. If not given, all classes are supposed to have weight one.

The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as `n_samples / (n_classes * np.bincount(y))`.

Note that these weights will be multiplied with sample_weight (passed through the fit method) if sample_weight is specified.

New in version 0.17: class_weight='balanced'

random_state : *int, RandomState instance, default=None*

Used when `solver` == 'sag', 'saga' or 'liblinear' to shuffle the data. See Glossary for details.

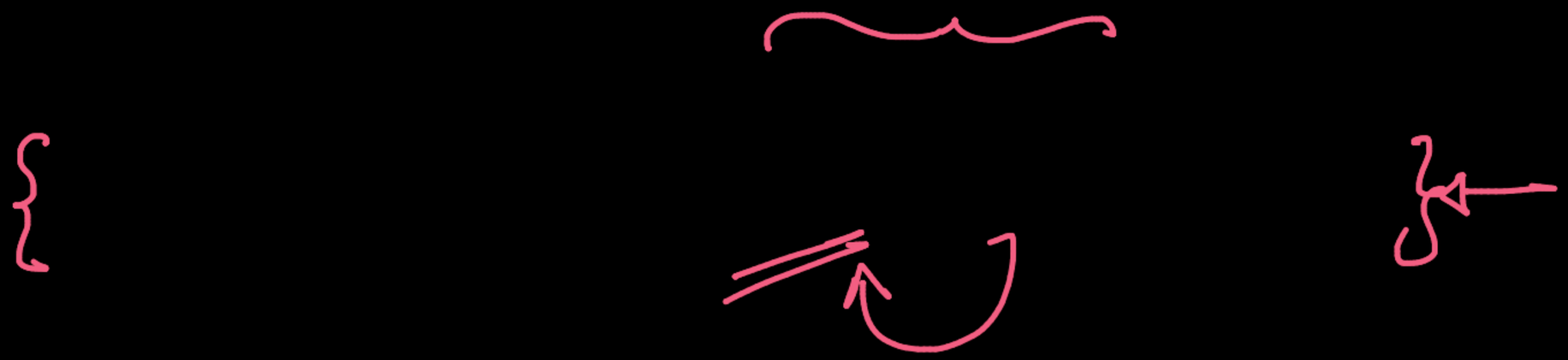
solver : *{'newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'}, default='lbfgs'*

Algorithm to use in the optimization problem. Default is 'lbfgs'. To choose a solver, you might want to consider the following aspects:

- For small datasets, 'liblinear' is a good choice, whereas 'sag' and 'saga' are faster for large ones;
- For multiclass problems, only 'newton-cg', 'sag', 'saga' and 'lbfgs' handle multinomial loss;
- 'liblinear' is limited to one-versus-rest schemes.

Warning: The choice of the algorithm depends on the penalty chosen: Supported penalties by solver:

Chrome File Edit View History Bookmarks Profiles Tab Window Help 23:28



Lr. Reg → Logistic Reg .....

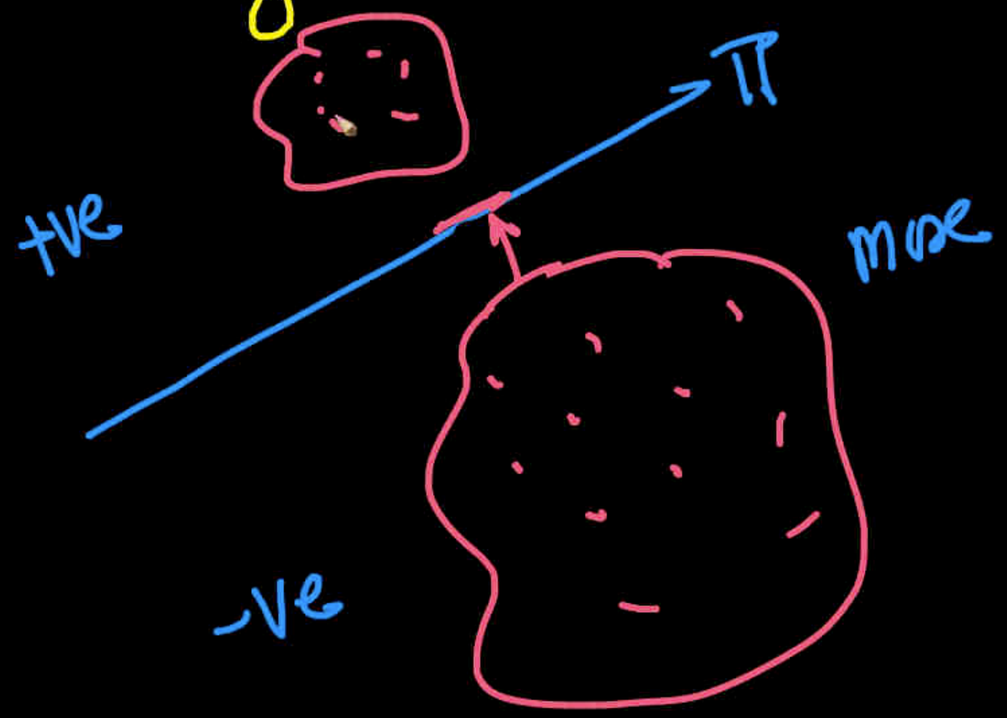
new-complx

92+4 ..... 96+3

Chrome File Edit View History Bookmarks Profiles Tab Window Help
77 / 78

Not - balancing

$$\sum_{i=1}^{n} \log \text{loss}_i$$



Chrome File Edit View History Bookmarks Profiles Tab Window Help 23:21

1 oversampling

+ve -ve

150 — 850

↓

900

2 i.e. weighting the loss fn $\sum_{i=1}^{n} \text{logloss}_i \cdot w_i$

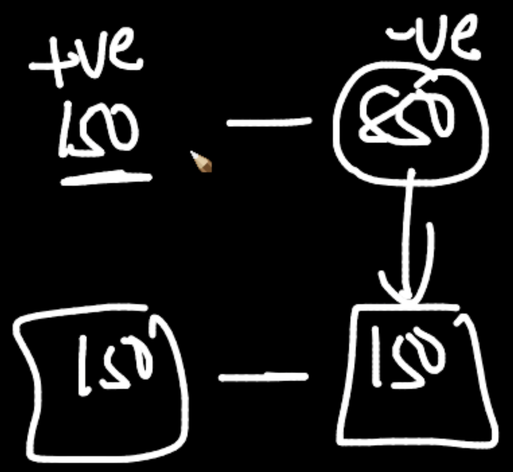
79 / 81

Chrome File Edit View History Bookmarks Profiles Tab Window Help 23:19

3 Undersampling

really large n

1 Billion



#datapoints ↓ ⇒ model sel ↓

80 / 80

Logistic Regr

- L_1 reg, elastic net reg
- log-loss vs squared loss (visually)
- pr, re, RoC, AuC

⋮

Logistic Regression.ipynb - Co | loss functions ml log loss vs | Loss Functions in Machine Le | plot(-log(1-x)) - Google Searc | sklearn.linear_model.LogisticR

colab.research.google.com/drive/1G_D39wa40kVahULehDS4IWhB_PXhP-xJ#scrollTo=OX5oPmT2Tuyc

+ Code + Text

RAM
Disk

```
#https://drive.google.com/file/d/1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_/view?usp=sharing

id = "1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_"
print("https://drive.google.com/uc?export=download&id=" + id)
```

```
https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_
```

```
[97] !wget "https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_" -O Churn.csv
```

```
--2022-04-27 16:57:18--  https://drive.google.com/uc?export=download&id=1Hryt6VSnHklyw3xxBG3nlhymzNAQhG-_
Resolving drive.google.com (drive.google.com)... 108.177.13.139, 108.177.13.101, 108.177.13.102, ...
Connecting to drive.google.com (drive.google.com)|108.177.13.139|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-0g-ag-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbp1/prghklqbqvn18o
Warning: wildcards not supported in HTTP.
--2022-04-27 16:57:19--  https://doc-0g-ag-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbp1
Resolving doc-0g-ag-docs.googleusercontent.com (doc-0g-ag-docs.googleusercontent.com)... 142.250.98.132, 2607:f8b0:4
Connecting to doc-0g-ag-docs.go                                                    ntent.com)|142.250.98.132|:443... con
HTTP request sent, awaiting res
Length: 289296 (283K) [text/csv]
Saving to: 'Churn.csv'

Churn.csv           100%[===================>] 282.52K  --.-KB/s    in 0.003s
```

82 / 82

0s completed at 23:01