

TOPICS

- Code for Decision Trees ✓ HPT
- Code for Random Forest
- Boosting
 - Intuition ✓
 - Gradient Boosting [Adaboost]
 - Bias-variance tradeoff.
 - Math dive-deep (Post-read videos)

GBDT

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=DymQNKTQODLB

+ Code + Text Reconnect

Business Problem + Data Processing

→ Google

[HR/People]

{x} # HR/People Analytics
import numpy as np
import pandas as pd
import io

[] !wget "https://drive.google.com/uc?export=download&id=16KtxSt_QEGQvfluEaMls5cCHPwhRXgCk" -O HR-Employee-Attrition.csv

```
--2022-05-13 13:24:44-- https://drive.google.com/uc?export=download&id=16KtxSt_QEGQvfluEaMls5cCHPwhRXgCk
Resolving drive.google.com (drive.google.com)... 142.250.141.113, 142.250.141.100, 142.250.141.139, ...
Connecting to drive.google.com (drive.google.com)|142.250.141.113|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-08-64-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbpl/157t46bjkl16kh
Warning: wildcards not supported in HTTP.
--2022-05-13 13:24:44-- https://doc-08-64-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbpl
Resolving doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleusercontent.com)... 142.250.141.132, 2607:f8b0:
Connecting to doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleusercontent.com)|142.250.141.132|:443... co
HTTP request sent, awaiting response... 200 OK
Length: 227977 (223K) [text/csv]
Saving to: 'HR-Employee-Attrition.csv'

HR-Employee-Attrition 100%[=====] 222.63K --.KB/s in 0.009s
```

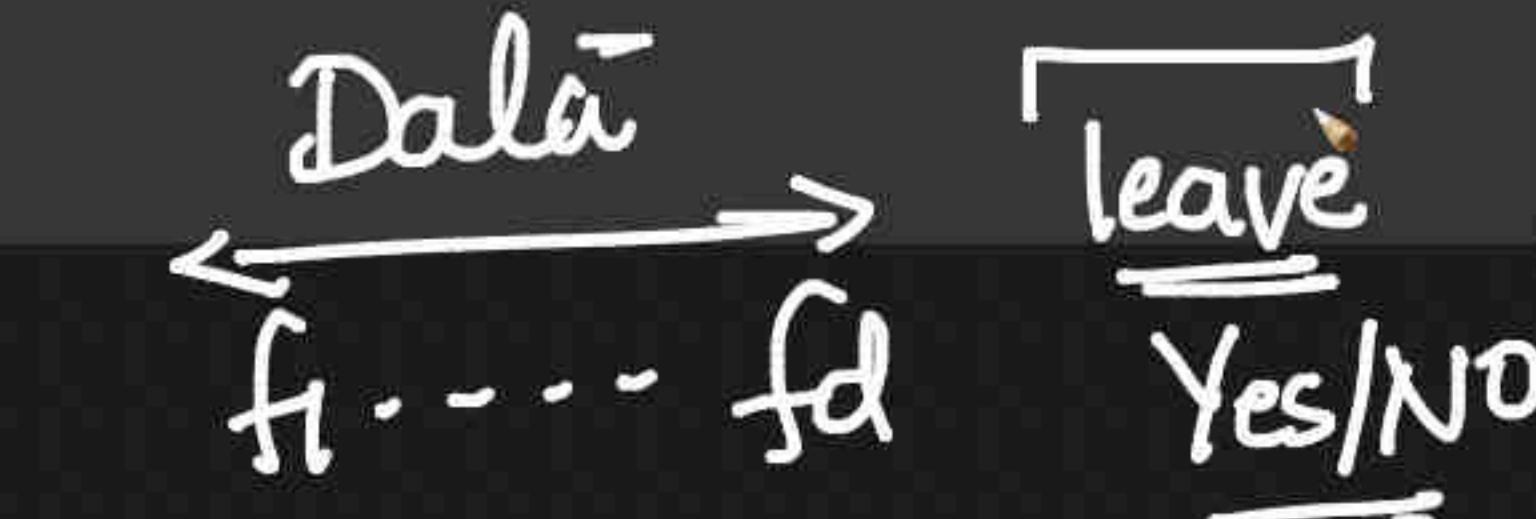
2022-05-13 13:24:44 124 7 MB/s 100% 222.63K / 227977 227977/227977

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=DymQNKTQODL8

+ Code + Text Reconnect

Business Problem + Data Processing



```
# HR/People Analytics
import numpy as np
import pandas as pd
import io
```

```
[ ] !wget "https://drive.google.com/uc?export=download&id=16KtxSt_QEGQvfluEaMls5cCHPwhRXgCk" -O HR-Employee-Attrition.csv
```

```
--2022-05-13 13:24:44-- https://drive.google.com/uc?export=download&id=16KtxSt_QEGQvfluEaMls5cCHPwhRXgCk
Resolving drive.google.com (drive.google.com)... 142.250.141.113, 142.250.141.100, 142.250.141.139, ...
Connecting to drive.google.com (drive.google.com)|142.250.141.113|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-08-64-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbpl/157t46bjkl16kh
Warning: wildcards not supported in HTTP.
--2022-05-13 13:24:44-- https://doc-08-64-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbpl
Resolving doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleusercontent.com)... 142.250.141.132, 2607:f8b0:
Connecting to doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleusercontent.com)|142.250.141.132|:443... co
HTTP request sent, awaiting response... 200 OK
Length: 227977 (223K) [text/csv]
Saving to: 'HR-Employee-Attrition.csv'
```

```
HR-Employee-Attriti 100%[=====] 222.63K --.KB/s in 0.009s
```

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=DymQNKTQODLB

+ Code + Text Reconnect

Business Problem + Data Processing

{x} # HR/People Analytics

import numpy as np
import pandas as pd
import io

[] !wget "https://drive.google.com/uc?export=download&id=16KtxSt_QEGQvfluEaMls5cCHPwhRXgCk" -O HR-Employee-Attrition.csv

--2022-05-13 13:24:44-- https://drive.google.com/uc?export=download&id=16KtxSt_QEGQvfluEaMls5cCHPwhRXgCk
Resolving drive.google.com (drive.google.com)... 142.250.141.113, 142.250.141.100, 142.250.141.139, ...
Connecting to drive.google.com (drive.google.com)|142.250.141.113|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-08-64-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbpl/157t46bjkl16kh
Warning: wildcards not supported in HTTP.
--2022-05-13 13:24:44-- https://doc-08-64-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc717deffksulhg5h7mbpl
Resolving doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleusercontent.com)... 142.250.141.132, 2607:f8b0:
Connecting to doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleusercontent.com)|142.250.141.132|:443... co
HTTP request sent, awaiting response... 200 OK
Length: 227977 (223K) [text/csv]
Saving to: 'HR-Employee-Attrition.csv'

HR-Employee-Attriti 100%[=====] 222.63K --.KB/s in 0.009s

2022-05-13 13:24:44 (224.7 MB/s)

4/4

```
DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | +  
colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=EOwBu11JOI5E  
+ Code + Text  
HR-Employee-Attrition 100% [=====] 444.0K ---.KB/S IN 0.009S  
[105]  
2022-05-13 13:24:44 (24.7 MB/s) - 'HR-Employee-Attrition.csv' saved [227977/227977]
```

{x}

df = pd.read_csv('HR-Employee-Attrition.csv')
df.head()

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	E
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	

5 rows × 35 columns



 DT and RF.ipynb - Colaborator ×  sklearn.tree.DecisionTreeClassi... ×  sklearn.ensemble.RandomFor

Code + Tex

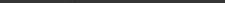
```
[221] df = pd.read_csv('HR-Employee-Attrition.csv')
      df.head()
```

Sch B M Phd ✓ RAM Disk

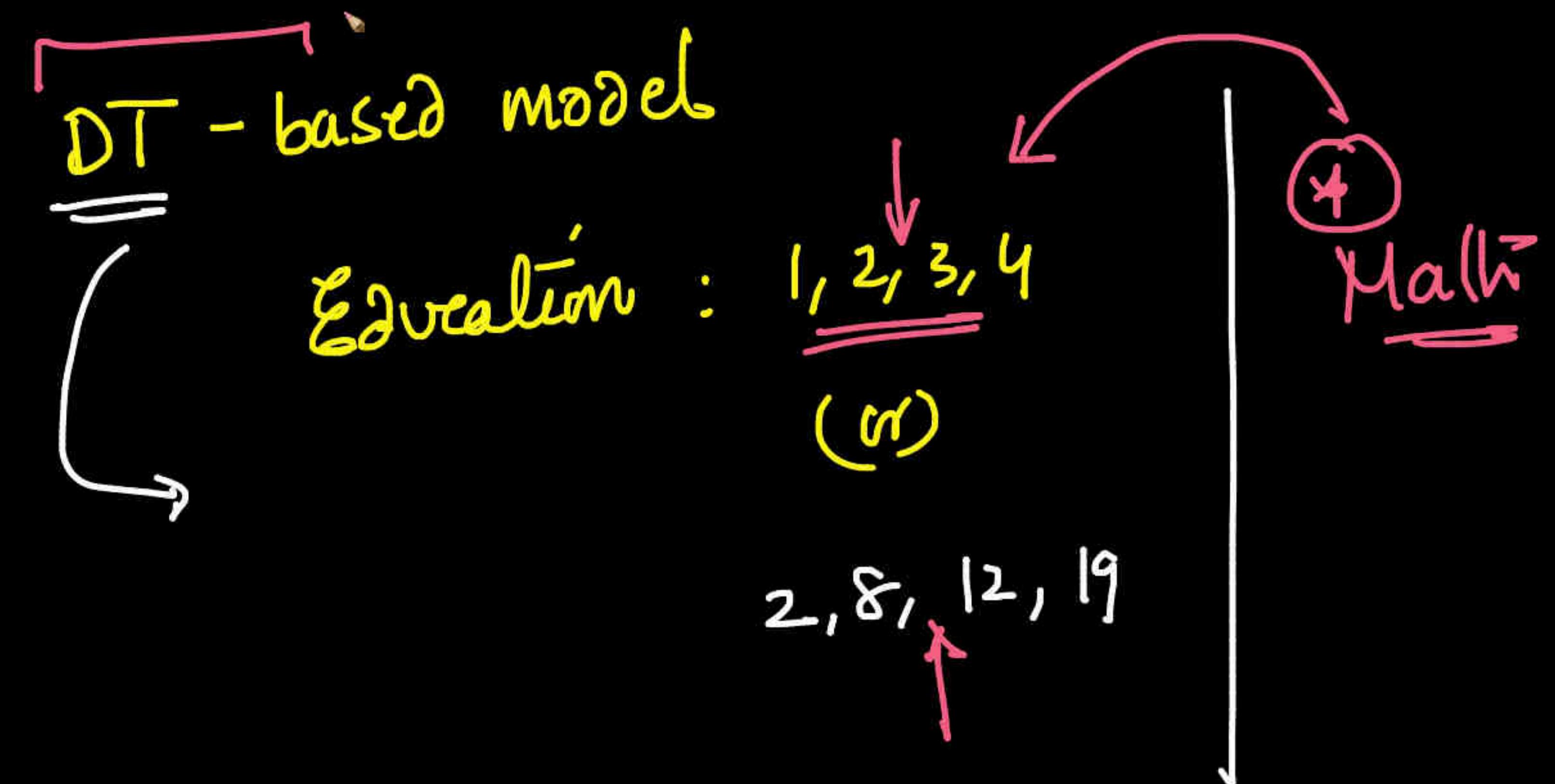
{x}

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	E
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	





df.info()



⊕ encoding = matters in logistic regression

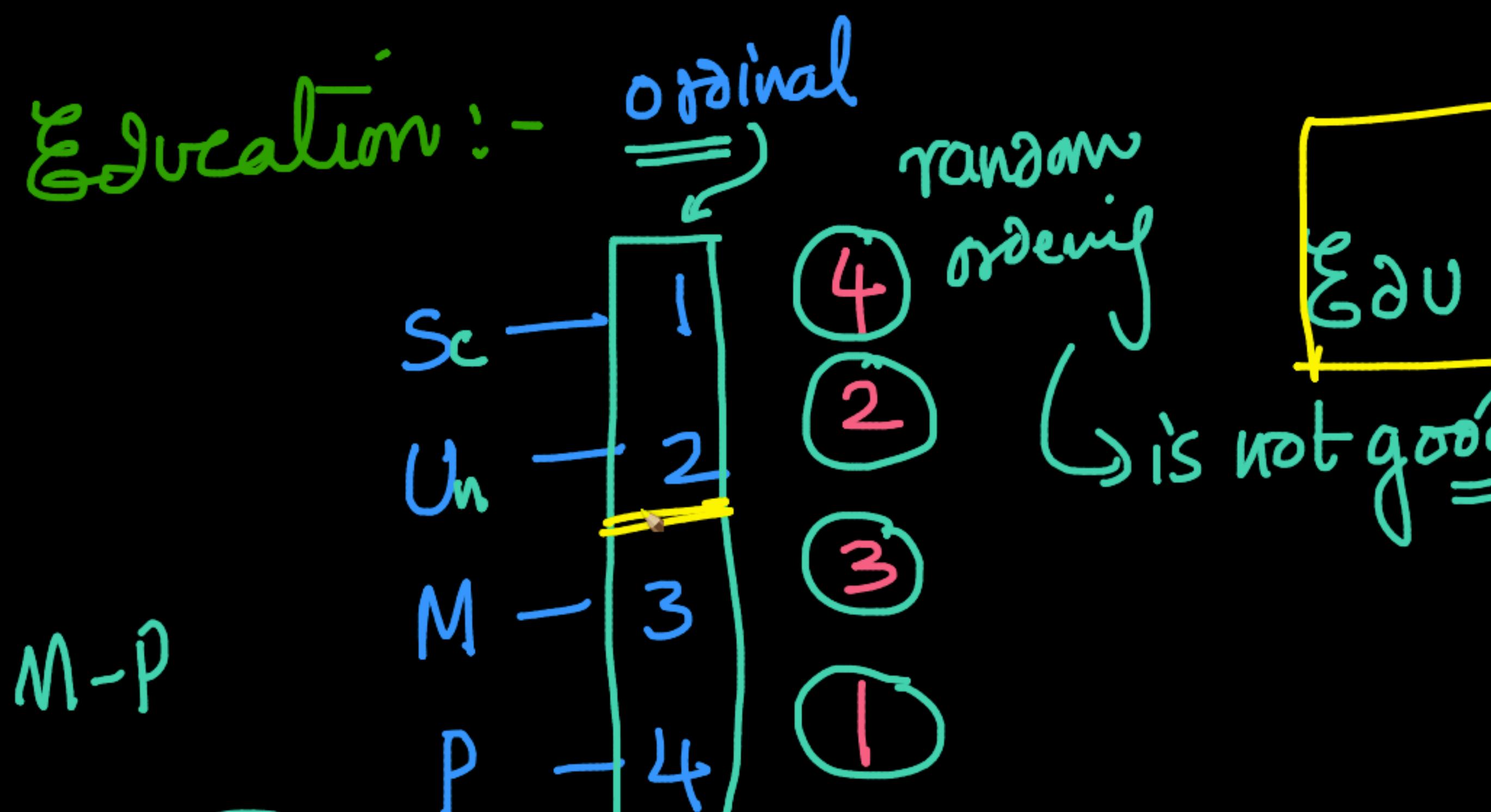
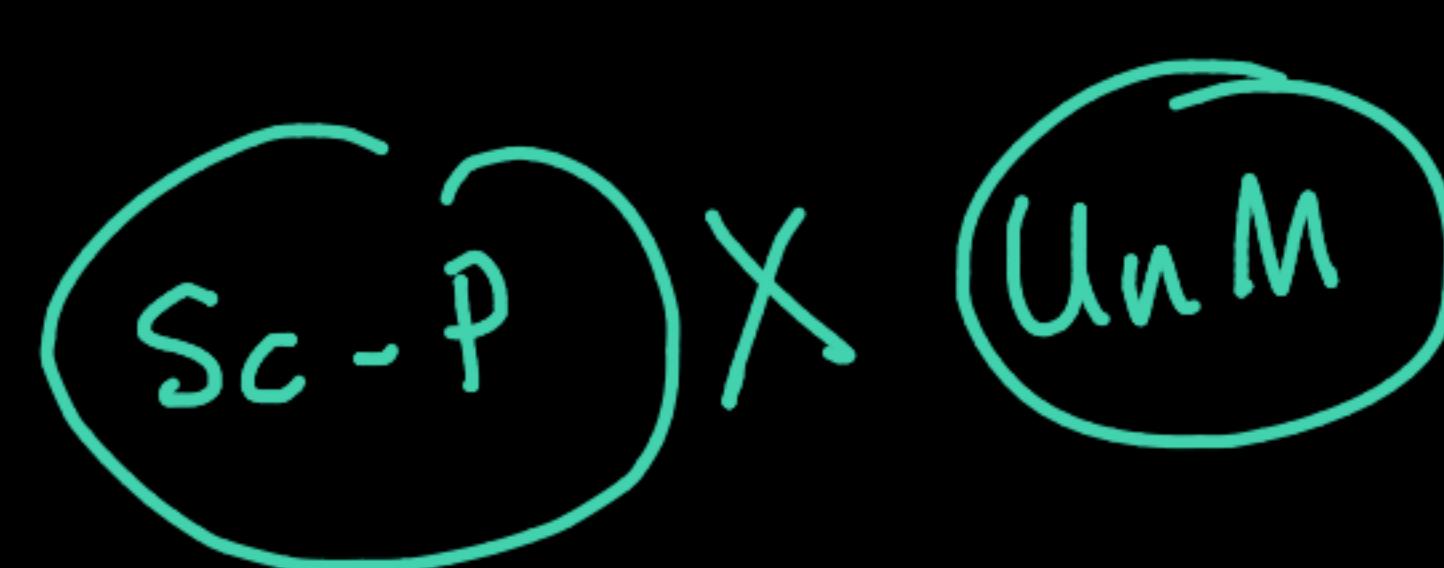
(distances)



scale matters

Sc - Un

M-P



Logistic regression

Evaluation → OHE ✓
→ Target Encoding ✓

 DT and RE ipynb - Collaborator sklearn.tree.DecisionTreeClass

sklearn.tree.DecisionTreeClass

sklearn.ensemble.RandomForest

+ Code + Text

RAM Disk

1

```
27 StandardHours          1470 non-null    int64
28 TotalWorkingYears      1470 non-null    int64
29 TrainingTimesLastYear 1470 non-null    int64
30 WorkLifeBalance        1470 non-null    int64
31 YearsAtCompany         1470 non-null    int64
32 YearsInCurrentRole    1470 non-null    int64
33 YearsSinceLastPromotion 1470 non-null    int64
34 YearsWithCurrManager   1470 non-null    int64
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

```
[108] # Business problems
# 1. Find employees who are at high risk of attrition and take remedial steps [ML Model]
# 2. Identify factors leading to attrition [Feature importance]
```

✓ [109] # Data preprocessing

```
# y_i's as 1 and 0 instead of Yes and No  
df['Attrition'].value_counts()  
# imbalanced data (~1:6 ratio): Need to account for this while building ML model
```

```
No      1233  
Yes     237  
Name: Attrition, dtype: int64
```

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=k-pOa2KB0v1B

+ Code + Text

✓ RAM Disk

Q {x} D:

TotalWorkingYears 1470 non-null int64
TrainingTimesLastYear 1470 non-null int64
WorkLifeBalance 1470 non-null int64
YearsAtCompany 1470 non-null int64
YearsInCurrentRole 1470 non-null int64
YearsSinceLastPromotion 1470 non-null int64
YearsWithCurrManager 1470 non-null int64
dtypes: int64(26), object(9)
memory usage: 402.1+ KB

[108] # Business problems
1. Find employees who are at high risk of attrition and take remedial steps [ML Model]
2. Identify factors leading to attrition [Feature importance]

[109] # Data preprocessing

y_i's as 1 and 0 instead of Yes and No
df['Attrition'].value_counts()
imbalanced data (~1:6 ratio): Need to account for this while building ML models

No 1233
Yes 237
Name: Attrition, dtype: int64

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=k-pOa2KB0v1B

+ Code + Text

✓ RAM Disk

Q {x} D

```
[27]: pd.read_csv('HR_comma_sep.csv')  
Out[27]:  
TotalWorkingYears      1470 non-null    int64  
TrainingTimesLastYear 1470 non-null    int64  
WorkLifeBalance        1470 non-null    int64  
YearsAtCompany         1470 non-null    int64  
YearsInCurrentRole    1470 non-null    int64  
YearsSinceLastPromotion 1470 non-null    int64  
YearsWithCurrManager   1470 non-null    int64  
dtypes: int64(26), object(9)  
memory usage: 402.1+ KB
```

[108] # Business problems
1. Find employees who are at high risk of attrition and take remedial steps [ML Model]
2. Identify factors leading to attrition [Feature importance]

[109] # Data preprocessing

y_i's as 1 and 0 instead of Yes and No
df['Attrition'].value_counts()
imbalanced data (~1:6 ratio): Need to account for this while building ML models

No 1233
Yes 237
Name: Attrition, dtype: int64

✓ [108] # Business problems
1. Find employees who are at high risk of attrition and take remedial steps [ML Model]
2. Identify factors leading to attrition [Feature importance]

```
# Data preprocessing  
# y_i's as 1 and 0 instead of Yes and No  
df['Attrition'].value_counts()  
# imbalanced data (~1:6 ratio): Need to account for this while building ML models
```

```
No      1233  
Yes     237  
Name: Attrition, dtype: int64
```

class-weights; SMOTE; ...

```
[110] df.loc[df['Attrition']=='Yes', 'Attrition'] = 1  
      df.loc[df['Attrition']=='No', 'Attrition'] = 0
```

✓ [111] df['Attrition'].value_counts()

```
0      1233  
1      237  
Name: Attrition, dtype: int64
```

 DT and RF.ipynb - Collaborator X | sklearn.tree.DecisionTreeClass X |

sklearn.tree.DecisionTreeClassifi

sklearn.ensemble.RandomFore x | +

colab.research.google.com/drive/1d2HJ9AKuIi7pwmDrL-2FOCncFubRF#scrollTo=Jdcg-fLA

 Update

+ Code + Text

A status bar at the bottom of the screen showing two icons: a green checkmark next to 'RAM' and a grey disk icon next to 'Disk'. The RAM icon has a progress bar above it.

1

```
# Data preprocessing  
  
# y_i's as 1 and 0 instead of Yes and No  
df['Attrition'].value_counts()  
# imbalanced data (~1:6 ratio): Need to account for this while building ML models
```

```
No      1233  
Yes     237  
Name: Attrition, dtype: int64
```

```
[110] df.loc[df[ 'Attrition' ]=='Yes', 'Attrition' ] = 1  
      df.loc[df[ 'Attrition' ]=='No', 'Attrition' ] = 0
```

[111] df['Attrition'].value_counts()

```
0      1233  
1      237  
Name: Attrition, dtype: int64
```

```
[112] # Sklearn DT does not handle categorical values natively.  
# Options: OHE or Target Encoding (our choice)  
# Converting all non numerical features into Target encoded features
```

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | +

```
[110] df.loc[df[ 'Attrition' ]=='Yes' , 'Attrition' ] = 1  
      df.loc[df[ 'Attrition' ]=='No' , 'Attrition' ] = 0
```

✓ [111] df['Attrition'].value_counts()

123

Name: Attrition, dtype: int64

[112] # Sklearn DT does not handle categorical values native

```
# Options: OHE or Target Encoding (our choice)
```

```
# Converting all non numerical features into Target encoded features
```

Q: Why not Label Encoding in Sklearn?

✓ [113] !pip install category-encoders

Requirement already satisfied: category-encoders in /usr/local/lib/python3.7/dist-packages (2.4.1)

Requirement already satisfied: scikit-learn>=0.20.0 in /usr/local/lib/python3.7/dist-packages (from category-encoder)

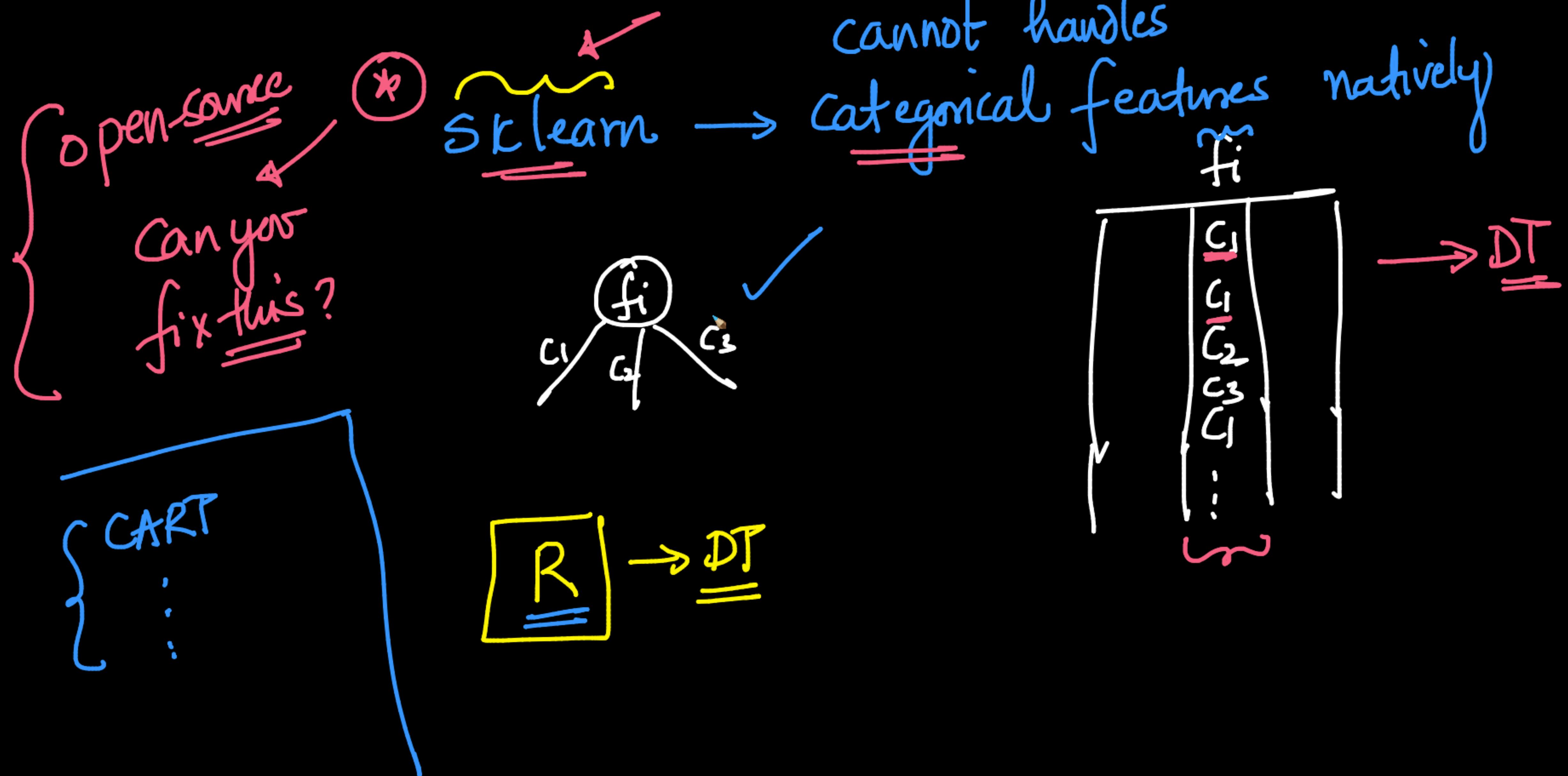
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.7/dist-packages (from category-encoders) (1.2

Requirement already satisfied: patsy>=0.5.1 in /usr/local/lib/python3.7/dist-packages (from category-encoders) (0.5.1)

Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.7/dist-packages (from category-encoders)

"Xgboost"

GBDT



f_i : Categorical - features for DT

↳ numeric

avoids → OHE - \times → explode #features =

if many categories → Target encoding $P(y_i=1 | f_i=c_1)$

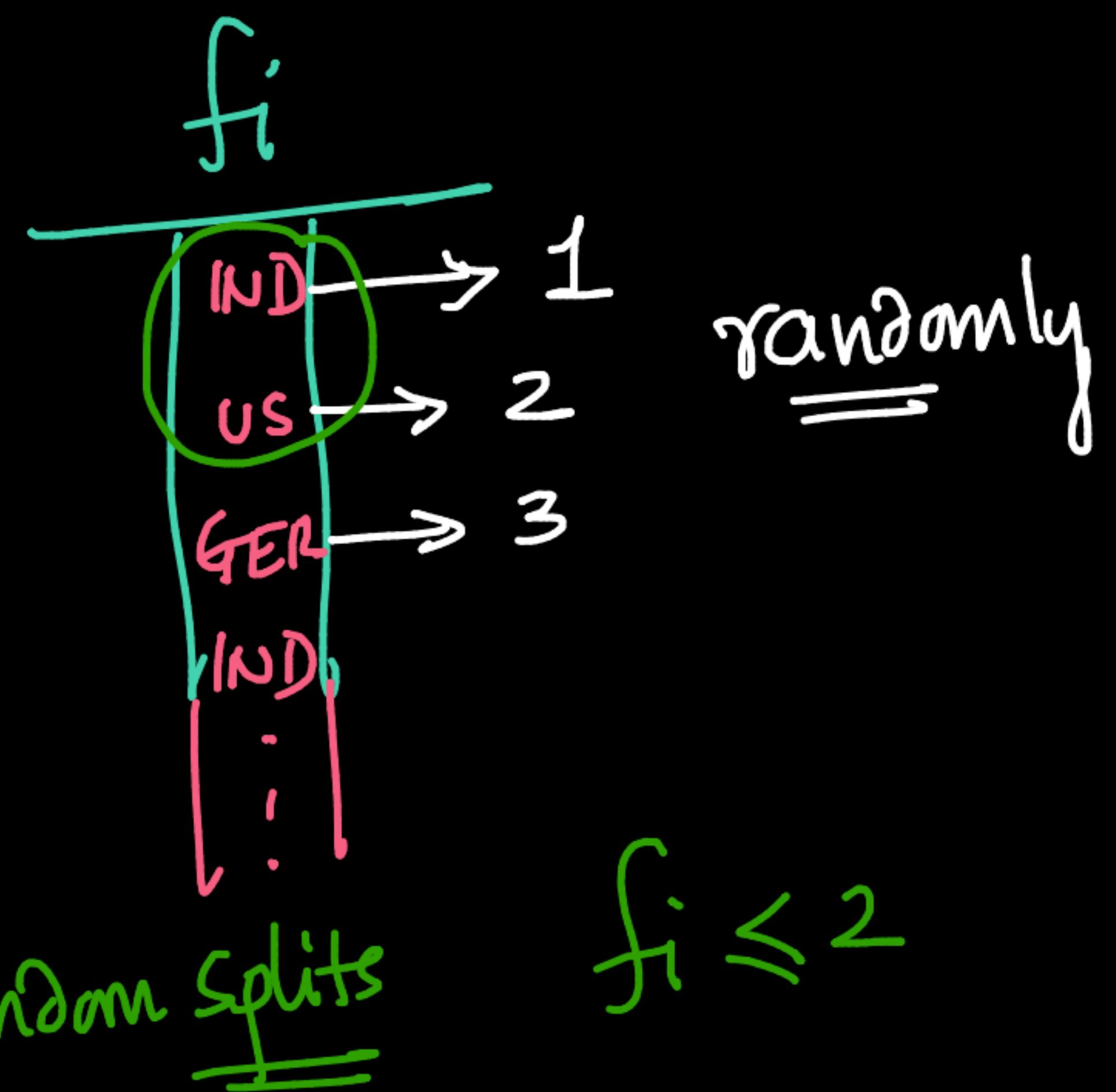
→ Label-encoding \times

Label-encoding:-

Country: non-ordinal

↓ creating random order

↳ random splits





Prev Up Next

scikit-learn 1.1.0

[Other versions](#)Please [cite us](#) if you use the software.

sklearn.preprocessing.LabelEncoder

sklearn.preprocessing.LabelEncoder

`class sklearn.preprocessing.LabelEncoder`[\[source\]](#)

Encode target labels with value between 0 and `n_classes-1`.

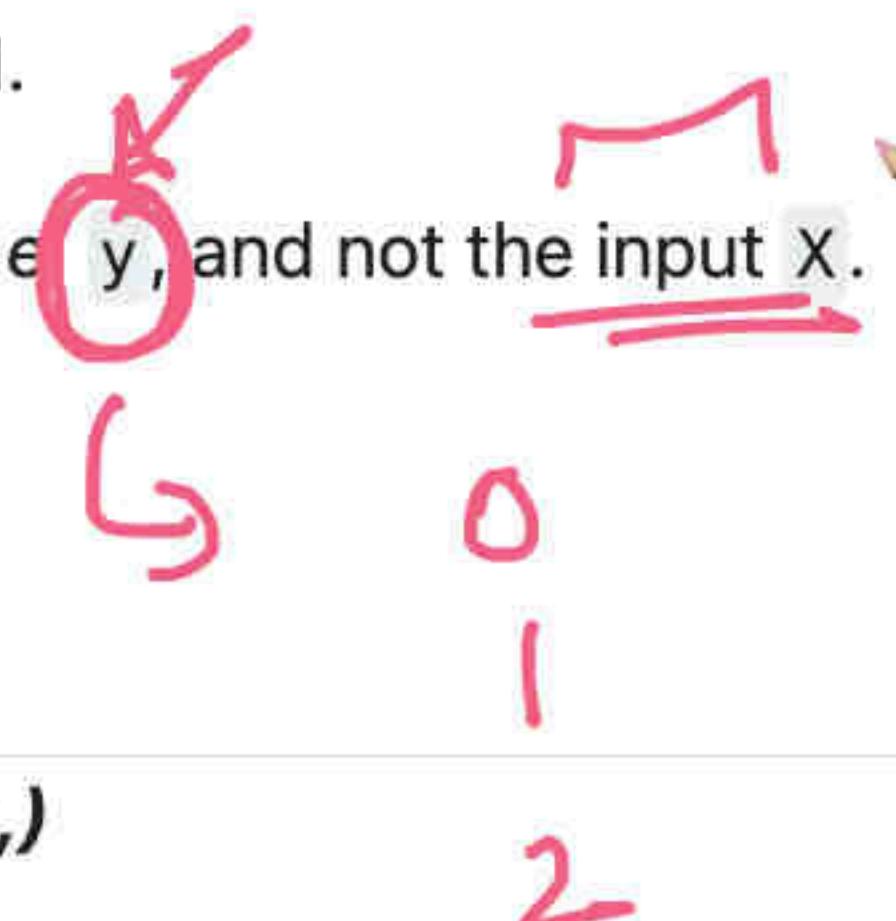
This transformer should be used to encode target values, i.e. y , and not the input X .

Read more in the [User Guide](#).

New in version 0.12.

Attributes: `classes_` : *ndarray of shape (n_classes,*

Holds the label for each class.



See also:

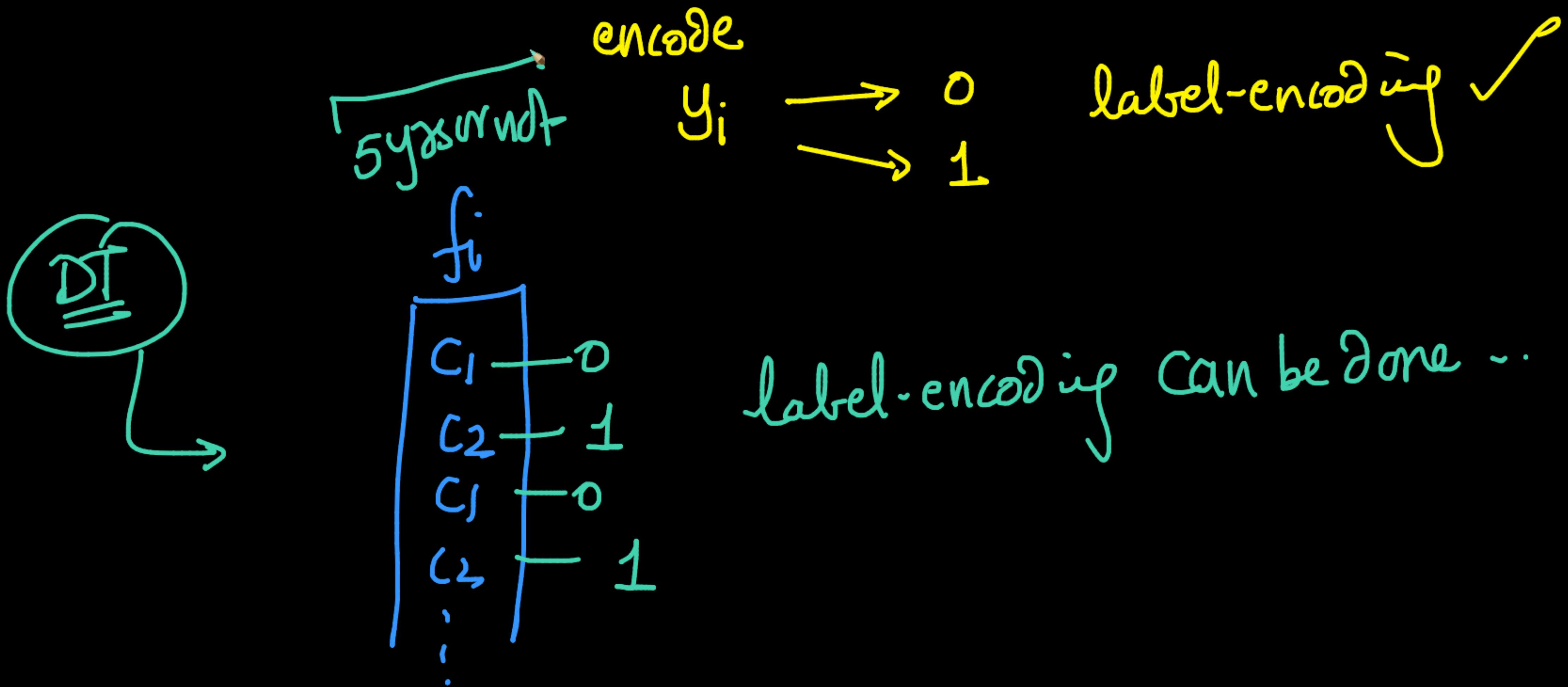
[OrdinalEncoder](#)

Encode categorical features using an ordinal encoding scheme.

[OneHotEncoder](#)

Encod...

{ \$ rm -rf * } lused



∞ DT and RF.ipynb - Colaboratory × sklearn.tree.DecisionTreeClass × sklearn.ensemble.RandomFore × sklearn.preprocessing.LabelEnc × +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=Oa6CGasjdbd8

+ Code + Text RAM Disk

Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from)

Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from patsy>=0.5.1->category-encoders)

Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca

{x}

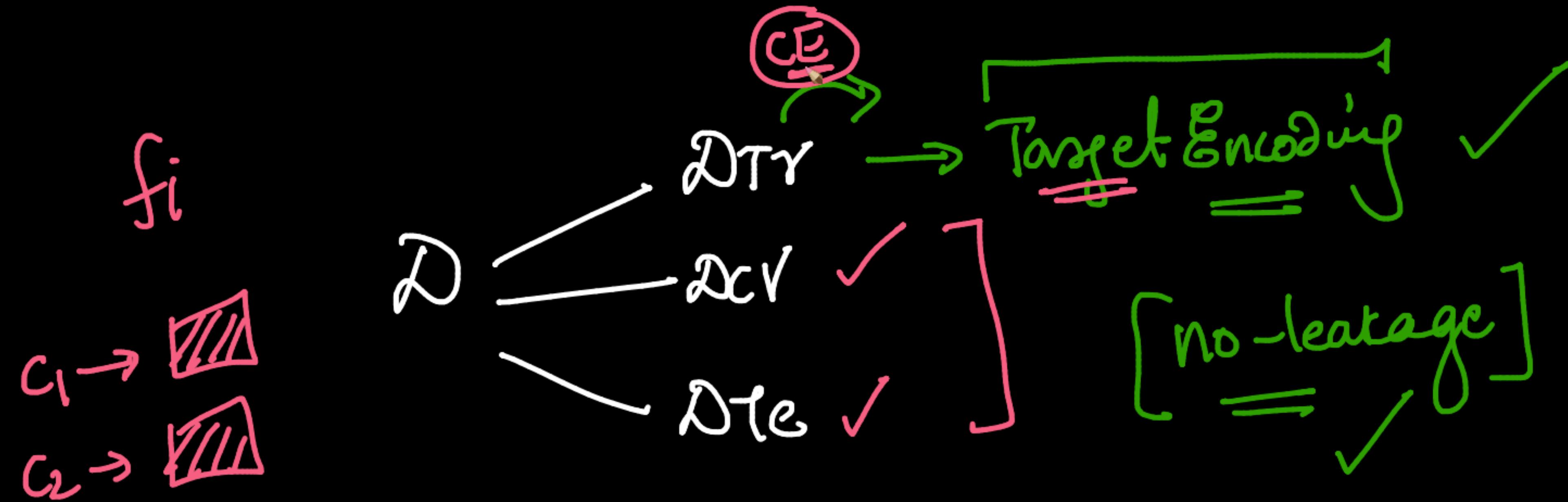
[114] from pandas.api.types import is_numeric_dtype
from category_encoders import TargetEncoder

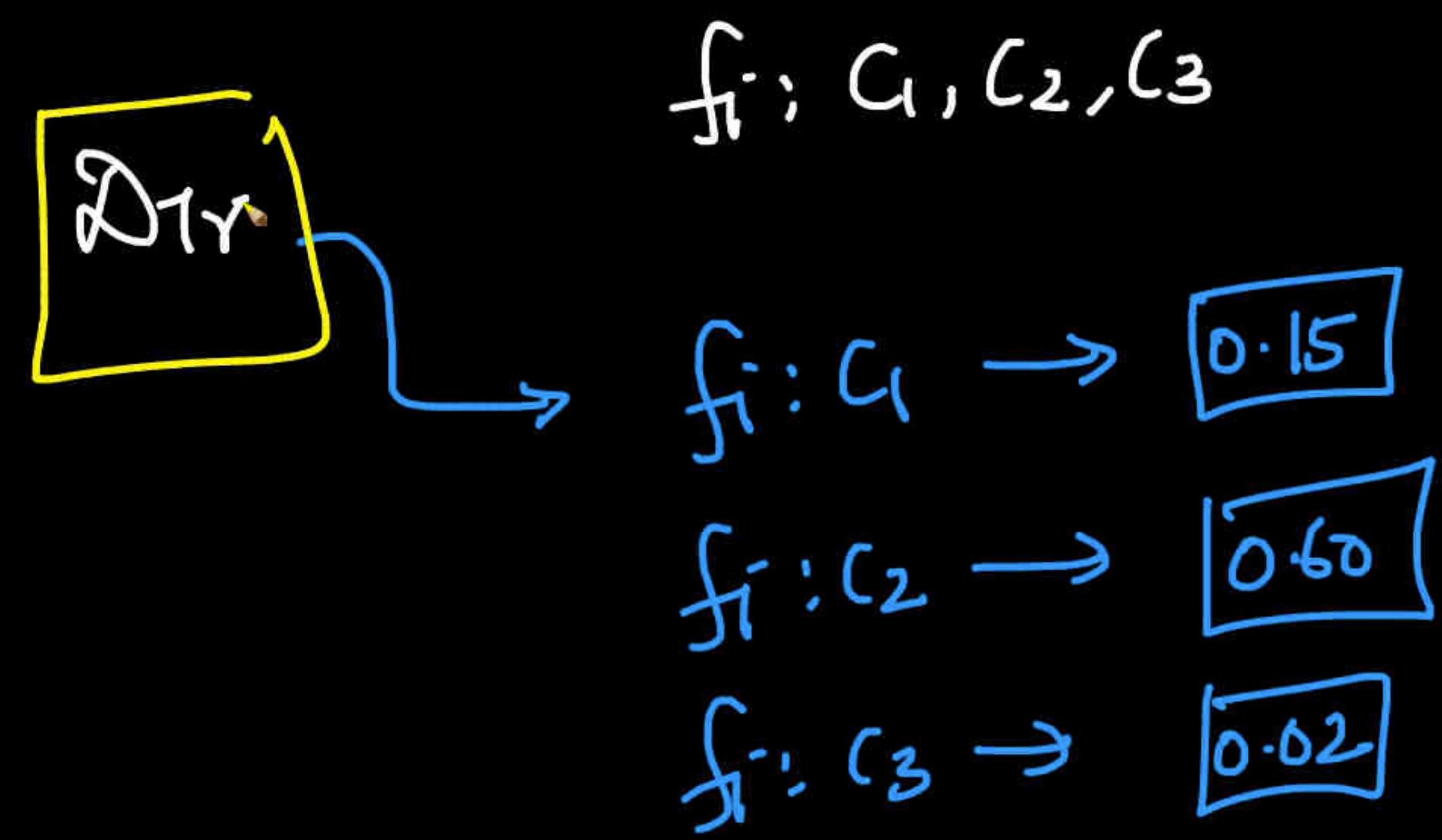
Convert all non-numeric features into Target-encoded features

HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:
 if is_numeric_dtype(df[col]) == False:
 encoder = TargetEncoder()
 df[col] = encoder.fit_transform(df[col], df['Attrition'])

[115] df.head()

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	F
0	41	1	0.149569	1102	0.206278		1	2	0.146865	1
1	49	0	0.249097	279	0.138398		8	1	0.146865	1





D_{CV} or D_{Te}

replace

$$\left. \begin{array}{l} f_1: C_1 \rightarrow 0.15 \\ f_1: C_2 \rightarrow 0.6 \\ f_1: C_3 \rightarrow 0.02 \end{array} \right\} \checkmark$$

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=0kvclMDBL1HW

+ Code + Text RAM Disk

Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca

Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca

{x} [113] HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:
 if is_numeric_dtype(df[col]) == False:
 print(col)

[114] from pandas.api.types import is_numeric_dtype
from category_encoders import TargetEncoder

Convert all non-numeric features into Target-encoded features
HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:
 if is_numeric_dtype(df[col]) == False:
 encoder = TargetEncoder()
 df[col] = encoder.fit_transform(df[col], df['Attrition'])

<>

[115] df.head()

RAM Disk

27/27

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEn... | ...

<https://colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrl-2EOCncEubBF#scrollTo=9kyvIMDBI1H>

Update :

+ Code + Text

A screenshot of the System Settings window in elementary OS. The window has a dark header bar with a green checkmark icon, the text "RAM" and "Disk", and a progress bar. Below the header are two sections: "RAM" and "Disk". Each section has a title, a progress bar, and a "Details" button. The "RAM" section shows a progress bar at approximately 75% completion. The "Disk" section shows a progress bar at approximately 50% completion. The bottom right corner of the window has a small checkmark icon.

```
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca
```

```
HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:
    if is_numeric_dtype(df[col]) == False:
        print(col)
```

BusinessTravel
Department
EducationField
Gender
JobRole
MaritalStatus
Over18
OverTime

-non-numeric

Editor: original

↓
numerical ✓

```
[114] from pandas.api.types import is_numeric_dtype
      from category_encoders import TargetEncoder

      # Convert all non-numeric features into Target-encoded features
      HR_col = list(df.columns)
      HR_col.remove('Attrition')
      for col in HR_col:
```

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=0kvclMDBL1HW

+ Code + Text

RAM Disk

Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca

Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca

{x} {x}

HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:
 if is_numeric_dtype(df[col]) == False:
 print(col)

BusinessTravel
Department
EducationField
Gender
JobRole
MaritalStatus
Over18
OverTime

DTY ✓ → Target encoding X

[114] from pandas.api.types import is_numeric_dtype
from category_encoders import TargetEncoder

Convert all non-numeric features into Target-encoded features
HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:

29 / 29

 DT and RF.ipynb - Colaboratory × sklearn.tree.DecisionTreeClass ×

sklearn.tree.DecisionTreeClass

sklearn.ensemble.RandomFor

sklearn.preprocessing.LabelEncoder

- Code + Text

✓ RAM Disk

```
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca
```

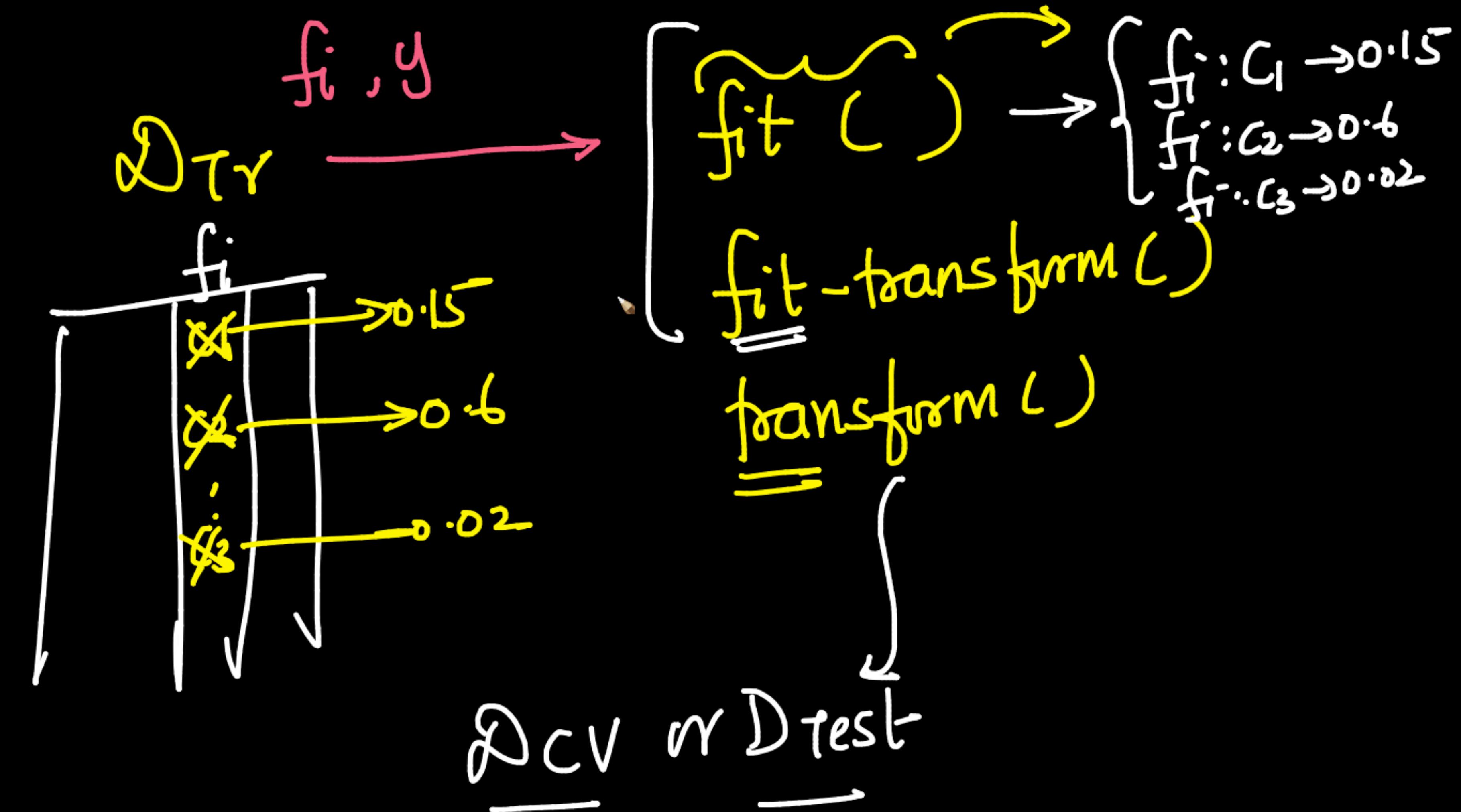
A horizontal bar containing several white icons on a dark background. From left to right, the icons are: an upward arrow, a downward arrow, a link symbol, a magnifying glass, a gear, a square with rounded corners, a trash can, and three vertical dots.

```
HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:
    if is_numeric_dtype(df[col]) == False:
        print(col)
```

BusinessTravel
Department
EducationField
Gender
JobRole
MaritalStatus
Over18
OverTime

```
[114] from pandas.api.types import is_numeric_dtype
      from category_encoders import TargetEncoder

      # Convert all non-numeric features into Target-encoded features
      HR_col = list(df.columns)
      HR_col.remove('Attrition')
      for col in HR_col:
```



contrib.scikit-learn.org/category_encoders/targetencoder.html

Backward Difference Coding

BaseN

Binary

CatBoost Encoder

Count Encoder

Generalized Linear Mixed Model Encoder

Hashing

Helmert Coding

James-Stein Encoder

Leave One Out

M-estimate

One Hot

Ordinal

Polynomial Coding

Sum Coding

Target Encoder

Weight of Evidence

Wrappers

Quantile Encoder

Summary Encoder

References

1 A Preprocessing Scheme for High-Cardinality Categorical Attributes in Classification and Prediction Problems, from <https://dl.acm.org/citation.cfm?id=507538>

Methods

<code>fit (X, y, **kwargs)</code>	Fit encoder according to X and y.
<code>fit_transform (X[, y])</code>	Encoders that utilize the target must make sure that the traini
<code>get_feature_names ()</code>	Returns the names of all transformed / added columns.
<code>get_params ([deep])</code>	Get parameters for this estimator.
<code>set_params (**params)</code>	Set the parameters of this estimator.
<code>transform (X[, y, override_return_df])</code>	Perform the transformation to new categorical data.

`fit_target_encoding`

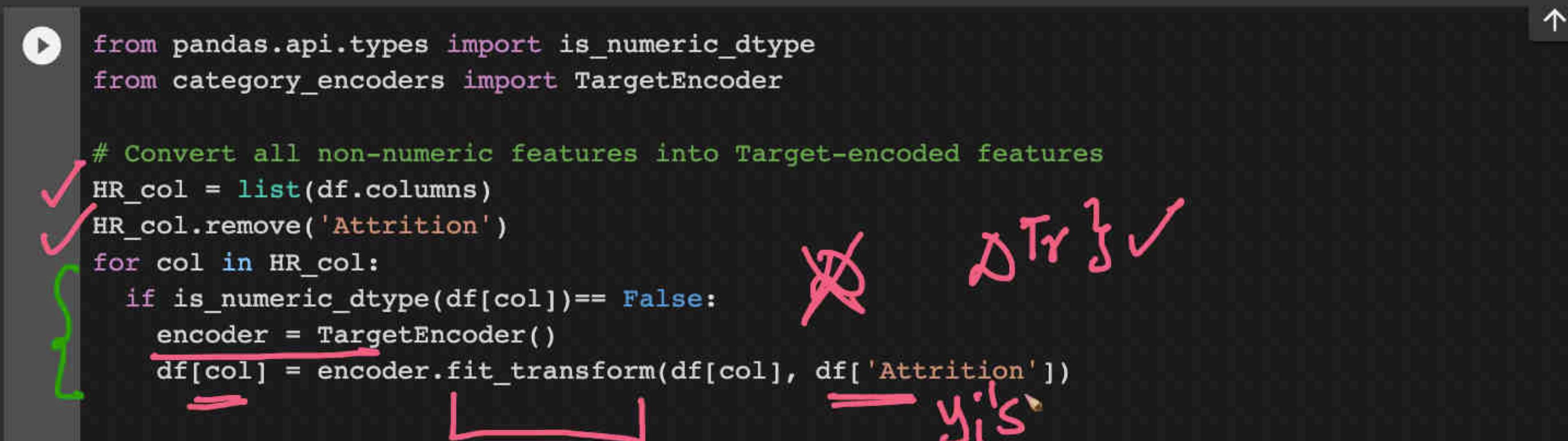
`target_encode`

`fit(X, y, **kwargs) [source]`

Fit encoder according to X and y.

+ Code + Tex

RAM Disk



✓ [115] df.head()

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	F
0	41	1		0.149569	1102	0.206278		1	2	0.146865
1	49	0		0.249097	279	0.138398		8	1	0.146865
2	37	1		0.149569	1373	0.138398		2	2	0.134146
3	33	0		0.249097	1392	0.138398		3	4	0.146865
4	27	0		0.149569	591	0.138398		2	1	0.135776

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=ObUfqzuweQ3X

+ Code + Text RAM Disk

RAM Disk

Code Editor

from pandas.api.types import is_numeric_dtype
from category_encoders import TargetEncoder

Convert all non-numeric features into Target-encoded features

HR_col = list(df.columns)
HR_col.remove('Attrition')
for col in HR_col:
 if is_numeric_dtype(df[col]) == False:
 encoder = TargetEncoder()
 df[col] = encoder.fit_transform(df[col], df['Attrition'])

[115] df.head()

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	F
0	41	1	0.149569	1102	0.206278		1	2	0.146865	1
1	49	0	0.249097	279	0.138398		8	1	0.146865	1
2	37	1	0.149569	1373	0.138398		2	2	0.134146	1
3	33	0	0.249097	1392	0.138398		3	4	0.146865	1
4	27	0	0.149569	591	0.138398		2	1	0.135776	1

+ Code + Text

✓ RAM	<div style="width: 100px; height: 10px; background-color: #ccc; border: 1px solid black;"></div>
Disk	<div style="width: 100px; height: 10px; background-color: #ccc; border: 1px solid black;"></div>



1

A set of small, light-gray navigation icons located at the bottom of the page. From left to right, they include: a double arrow pointing up and down, a circular arrow, a speech bubble, a gear, a square with a diagonal line, a trash can, and three vertical dots.

8

```
#Check if any non-numeric column exists in the DF  
for col in HR_col:  
    if is_numeric_dtype(df[col]) == False:  
        print(col)
```

```
# Remove useless columns  
df['EmployeeNumber'].value_counts()
```

```
{ 1      1  
1391   1  
1389   1  
1387   1  
1383   1  
       ..  
659    1  
657    1  
656    1  
655    1  
2068   1  
Name: EmployeeNumber, Length: 1470, dtype: int64
```

[118] df['EmployeeCount'].value_counts()

1 1470

 DT and RF.ipynb - Colaborator | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEn... | Target Encoder – Category En...

```
- Code + Text

      1      1
[117] 1391      1
      1389      1
      1387      1
      1383      1
      ..
      659      1
      657      1
      656      1
      655      1
      2068      1
Name: EmployeeNumber, Length: 1470, dtype: int64
```

```
df[ 'EmployeeCount' ].value_counts()
```

1 1470

Name: EmployeeCount, dtype: int64

[119] df['StandardHours'].value_counts()

80 1470

Name: StandardHours, dtype: int64

✓ [120] df['Over18'].value_counts()

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder — Category Encoders

colab.research.google.com/drive/1d2HJ9AKuIif83i7pwmDrL-2FOCncFubRF#scrollTo=-pCaD2hch39K

Update

✓ RAM Disk	<div style="width: 100px; height: 10px; background-color: #ccc; border: 1px solid black;"></div>
✗ RAM Disk	<div style="width: 100px; height: 10px; background-color: #ccc; border: 1px solid black;"></div>

```
✓ [118] df['EmployeeCount'].value_counts()
```

```
1      1470  
Name: EmployeeCount, dtype: int64
```

```
[119] df['StandardHours'].value_counts()
```

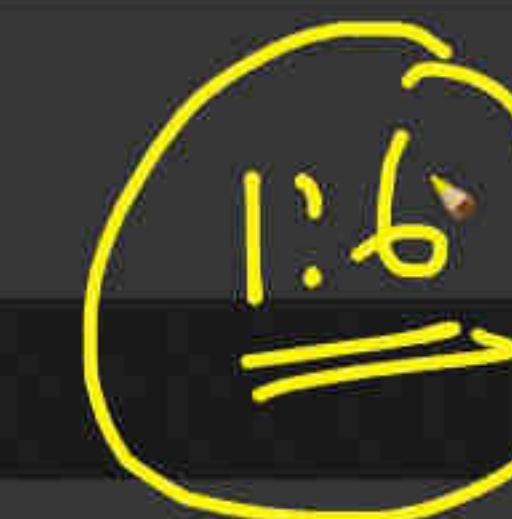
```
80      1470  
Name: StandardHours, dtype: int64
```

```
✓ ⏎ df['Over18'].value_counts()
```

```
0.161224      1470  
Name: Over18, dtype: int64
```

```
[121] y = df['Attrition']
      y = y.astype('int')
```

```
[122] df.drop(['Attrition', 'EmployeeCount', 'EmployeeNumber',
    'StandardHours', 'Over18'], axis=1, inplace=True)
    print('Size of Full dataset is: {}'.format(df.shape))
```



 DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEn... | Target Encoder – Category En... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=-pCaD2hch39K

◎ 书 猫 四 月 Update

+ Code + Text

- ✓ RAM Disk

V

```
✓ [118] df['EmployeeCount'].value_counts()
```

```
1      1470  
Name: EmployeeCount, dtype: int64
```

```
[119] df['StandardHours'].value_counts()
```

```
80      1470  
Name: StandardHours, dtype: int64
```

 df['Over18'].value_counts()

```
0.161224      1470  
Name: Over18, dtype: int64
```

```
[121] y = df['Attrition']
      y = y.astype('int')
```

```
[122] df.drop(['Attrition', 'EmployeeCount', 'EmployeeNumber',
    'StandardHours', 'Over18'], axis=1, inplace=True)
print('Size of Full dataset is: {}'.format(df.shape))
```

 DT and RF.ipynb - Colaborator | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder – Category Encoding

+ Code + Text

```
[120]: df['Over18'].value_counts()
```

```
0.161224      1470  
Name: Over18, dtype: int64
```

```
y = df['Attrition']  
y = y.astype('int')
```

```
✓ [122] df.drop(['Attrition', 'EmployeeCount', 'EmployeeNumber',
  Is           'StandardHours', 'Over18'], axis=1, inplace=True
        print('Size of Full dataset is: {}'.format(df.shape))
```

x = df

Size of Full dataset is: (1470, 30)

```
[123] # 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 =
```



 DT and RF.ipynb - Colaborator | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder – Category Encoding

+ Code + Text

```
[120]: df['Over18'].value_counts()
```

```
0.161224      1470  
Name: Over18, dtype: int64
```

```
y = df['Attrition']
y = y.astype('int')
```

```
[122] df.drop(['Attrition', 'EmployeeCount', 'EmployeeNumber',
    'StandardHours', 'Over18'], axis=1, inplace=True)
print('Size of Full dataset is: {}'.format(df.shape))
```

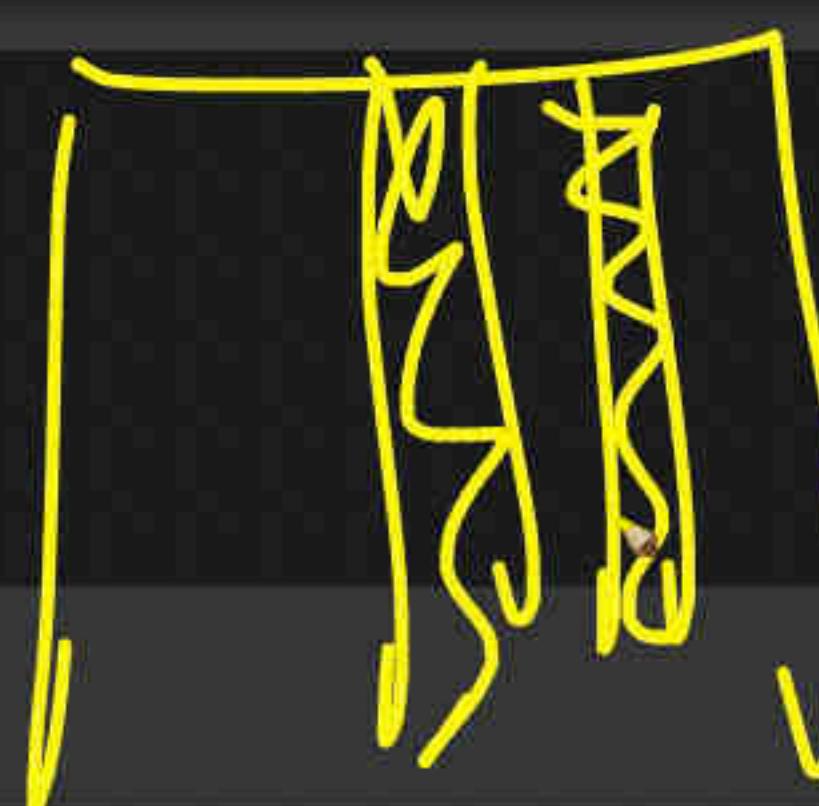
x = df

Size of Full dataset is: (1470, 30)

```
[123] # 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 =
```



∞ DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=fD_-6k16hPnT

+ Code + Text

RAM Disk

✓ [120] df['Over18'].value_counts()

0.161224 1470
Name: Over18, dtype: int64

{x}

✓ [121] y = df['Attrition']
y = y.astype('int')

✓ [122] df.drop(['Attrition', 'EmployeeCount', 'EmployeeNumber',
'StandardHours', 'Over18'], axis=1, inplace=True)
print('Size of Full dataset is: {}'.format(df.shape))

X = df

Size of Full dataset is: (1470, 30)

✓ [123] # 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 =

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=fD_-6k16hPnT

+ Code + Text

[123] # Train, CV, test split
from sklearn.model_selection import train_test_split
#0.6, 0.2, 0.2 split

{x}
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 =

[124] print("Number transactions X_train dataset: ", X_train.shape)
print("Number transactions y_train dataset: ", y_train.shape)
print("Number transactions X_CV dataset: ", X_val.shape)
print("Number transactions y_CV dataset: ", y_val.shape)
print("Number transactions X_test dataset: ", X_test.shape)
print("Number transactions y_test dataset: ", y_test.shape)

Number transactions X_train dataset: (882, 30)
Number transactions y_train dataset: (882,)
Number transactions X_CV dataset: (294, 30)
Number transactions y_CV dataset: (294,)
Number transactions X_test dataset: (294, 30)
Number transactions y_test dataset: (294,)

▶ Decision Trees

Chrome File Edit View History Bookmarks Profiles Tab Window Help

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=fD_-6k16hPnT

+ Code + Text RAM Disk Update

[123] # 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 =

[124] print("Number transactions X_train dataset: ", X_train.shape)
print("Number transactions y_train dataset: ", y_train.shape)
print("Number transactions X_CV dataset: ", X_val.shape)
print("Number transactions y_CV dataset: ", y_val.shape)
print("Number transactions X_test dataset: ", X_test.shape)
print("Number transactions y_test dataset: ", y_test.shape)

Number transactions X_train dataset: (882, 30)
Number transactions y_train dataset: (882,)
Number transactions X_CV dataset: (294, 30)
Number transactions y_CV dataset: (294,)
Number transactions X_test dataset: (294, 30)
Number transactions y_test dataset: (294,)

Decision Trees

DT and RF.ipynb - Colaborator | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder – Category Encoders

- Code + Text

✓ RAM Disk



```
# Hyper-pram tuning + DT model
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import f1_score

train_scores = []
val_scores = []

l=1
u=20
d=1
w=1.0

for depth in np.arange(l,u,d):
    clf = DecisionTreeClassifier(random_state=0, max_depth=depth, class_weight={ 0:0.1, 1:w } )
    clf.fit(X_train, y_train)
    train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val_score)
```

✓ [209] import matplotlib.pyplot as plt

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder — Category Encoders

<https://colab.research.google.com/drive/1d2H19AKuJf83i7nwMDd-2EOGncEuBRE#scrollTo=Y2vZAEiWc-8E>

Update

+ Code + Text

- ✓ RAM Disk

1

A set of small, light-gray navigation icons located at the bottom of the page. From left to right, they include: an upward arrow, a downward arrow, a link icon (a circle with a line), a message icon (a speech bubble), a gear icon (a settings gear), a refresh/circular arrow icon, a trash can icon, and three vertical dots representing more options.

8

```
# Hyper-pram tuning + DT model
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import f1_score

train_scores = []
val_scores = []

l=1
u=20
d=1
w=1.0

for depth in np.arange(l,u,d):
    clf = DecisionTreeClassifier(random_state=0, max_depth=depth, class_weight={ 0:0.1, 1:w } )
    clf.fit(X_train, y_train)
    train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val_score)

    print("Depth: ", depth, " Train Score: ", train_score, " Val Score: ", val_score)
```

$l, 2, 3, 4, \dots, 20$

l 2, 3, 4, . . . 27

$$\left\{ \begin{array}{l} l=1 \\ u=20 \\ d=1 \\ w=1. \end{array} \right.$$

1
20

✓ [209] import matplotlib.pyplot as plt

 DT and RF.ipynb - Colaborator |  sklearn.tree.DecisionTreeClass... |  sklearn.ensemble.RandomFore... |  sklearn.preprocessing.LabelEr... |  Target Encoder – Category En...

<https://colab.research.google.com/drive/1d2H19AKUfF83iZpwmDrl-2EDGncEUDRE#scrollTo=x2yZAEiWc-8E>

Update

+ Code + Tex

RAM Disk

1



6

```
# Hyper-pram tuning + DT model
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import f1_score

train_scores = []
val_scores = []

l=1
u=20
d=1
w=1.0

for depth in np.arange(l,u,d):
    clf = DecisionTreeClassifier(random_state=0,
        clf.fit(X_train, y_train)
    train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val_score)
```

depth = 1, 2, ..., 2D

hyper-param

✓ [209] import matplotlib.pyplot as plt

Combinations

{ depth = 1
depth = 1
"
depth = 2
"
}

class weights: 0.1 : 0.6

: 0.1 : 1.0

: 0.1 : 2.0

" C.W = 0.1 : 0.6

" 0.1 : 1.0

,
,

GBDT → automatic

GridSearch
Random Search
on HP =

 DT and RF.ipynb - Colaborator |  sklearn.tree.DecisionTreeClass... |  sklearn.ensemble.RandomFore... |  sklearn.preprocessing.LabelEr... |  Target Encoder – Category En...

- Code + Text

RAM Disk

1=1

$$u=20$$

d=

w=1.

```
for depth in np.arange(l,u,d):
```

```
✓clf = DecisionTreeClassifier(random_state=0, max_depth=depth, class_weight={ 0:0.1, 1:w } )
```

```
clf.fit(X_train, y_train)
```

```
train_y_pred = clf.predict(X_train)
```

```
val y_pred = clf.predict(X_val)
```

```
train score = f1 score(y train, train y pred)
```

```
val score = f1_score(y_val, val_y_pred)
```

```
train_scores.append(train_score)
```

```
val scores.append(val score)
```

10

→ tunable

[209] import matplotlib.pyplot as plt

```
plt.figure(
```

```
plt.plot(list(np.arange(l,u,d)), train scores, label="train")
```

```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
```

```
plt.legend(loc='lower right')
```

```
plt.xlabel("lambda")
```

```
plt.ylabel("F1-Score")
```

```
plt.grid()
```

 DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEn... | Target Encoder – Category En... | +

- Code + Text

- ✓ RAM Disk



1=1

$$u=20$$

d=1

w=1,0

```
for depth in np.arange(l,u,d):
```

```
clf = DecisionTreeClassifier(random_state=0, max_depth=depth, class_weight={ 0:0.1, 1:w } )
```

```
clf.fit(X_train, y_train)
```

```
train_y_pred = clf.predict(X_train)
```

```
val_y_pred = clf.predict(X_val)
```

```
train_score = f1_score(y_train, train_y_pred)
```

```
val_score = f1_score(y_val, val_y_pred)
```

```
train_scores.append(train_score)
```

```
val_scores.append(val_score)
```

✓ [209] import matplotlib.pyplot as plt

```
plt.figure()
```

```
plt.plot(list(np.arange(l,u,d))), train_scores, label="train"
```

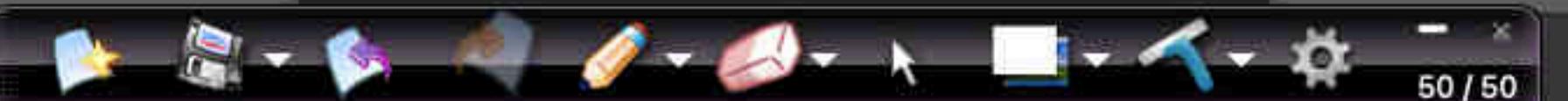
```
plt.plot(list(np.arange(l,u,d))), val_scores, label="val")
```

```
plt.legend(loc='lower right')
```

```
plt.xlabel("lambda")
```

```
plt.ylabel("F1-Score")
```

```
plt.grid()
```



 DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEn... | Target Encoder – Category En... | +

colab.research.google.com/drive/1d2HJ9AKuif83i7pwmDrL-2FOCncFubRF#scrollTo=X2vZAFiW

Update ::

+ Code + Text

- ✓ RAM Disk

1

```
l=1
u=20
d=1
w=1.0

for depth in np.arange(l,u,d):
    clf = DecisionTreeClassifier(random_state=0, max_depth=depth, class_weight={ 0:0.1, 1:w } )
    clf.fit(X_train, Y_train)
    ✓train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val score)
```

✓ [209] import matplotlib.pyplot as plt

```
plt.figure()
plt.plot(list(np.arange(l,u,d)), train_scores, label="train")
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
plt.ylabel("F1-Score")
plt.grid()
```

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder — Category Encoders

colab.research.google.com/drive/1d2HJ9AKuIif83i7pwmDrL-2FOCncFubRF#scrollTo=X2vZAFiWc--

Update

+ Code + Text

RAM Disk

1

```
l=1  
u=20  
d=1  
w=1.0  
  
for depth in np.arange(l,u,d):  
    clf = DecisionTreeClassifier(random_state=0, max_depth=depth, class_weight={ 0:0.1, 1:w } )  
    clf.fit(X_train, y_train)  
    train_y_pred = clf.predict(X_train)  
    val_y_pred = clf.predict(X_val)  
    train_score = f1_score(y_train, train_y_pred)  
    val_score = f1_score(y_val, val_y_pred)  
    train_scores.append(train_score)  
    val_scores.append(val_score)
```

✓ [209] import matplotlib.pyplot as plt

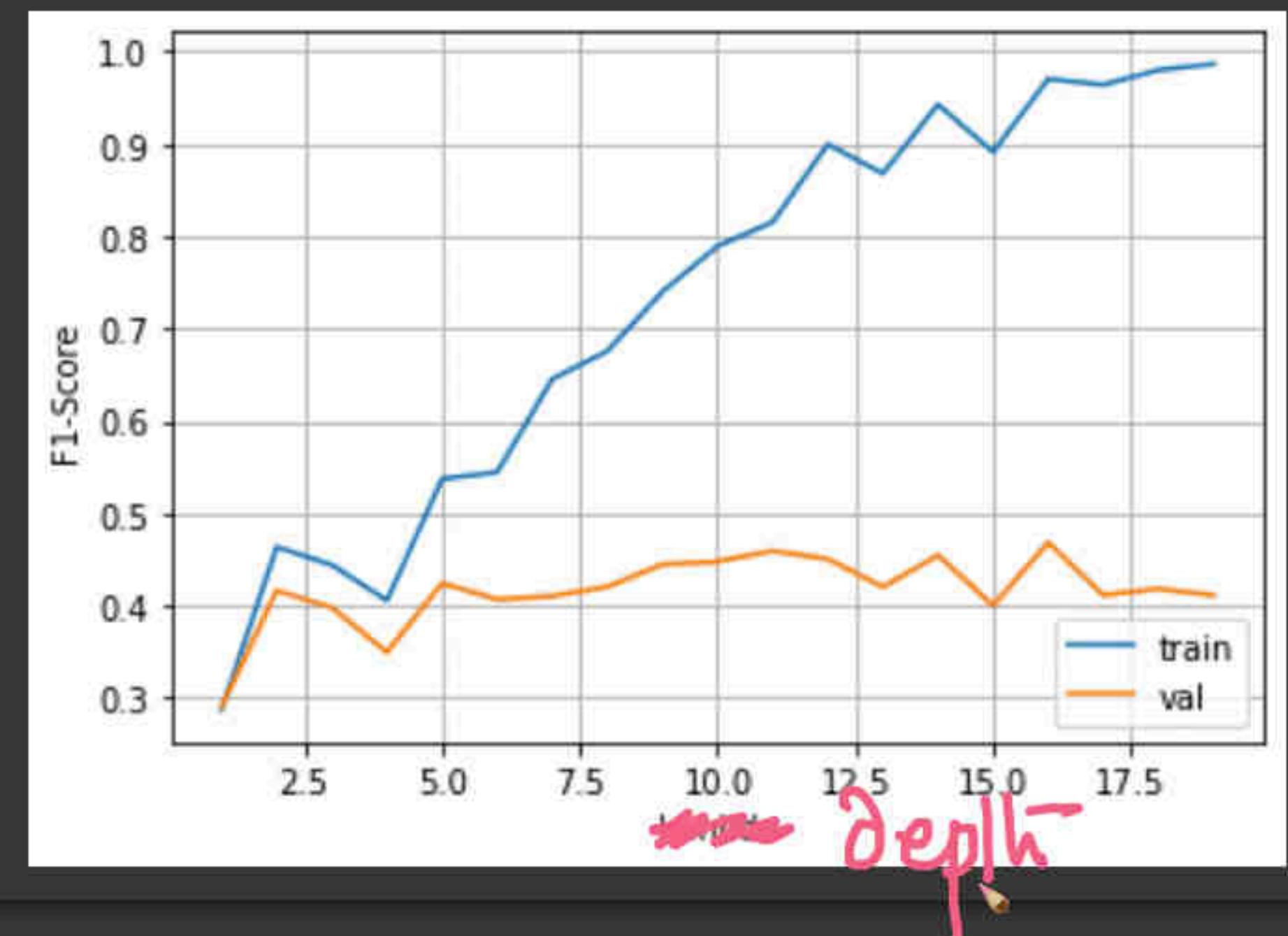
```
plt.figure()
plt.plot(list(np.arange(l,u,d)), train_scores, label="train")
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
plt.ylabel("F1-Score")
plt.grid()
```

+ Code + Text

✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("lambda")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



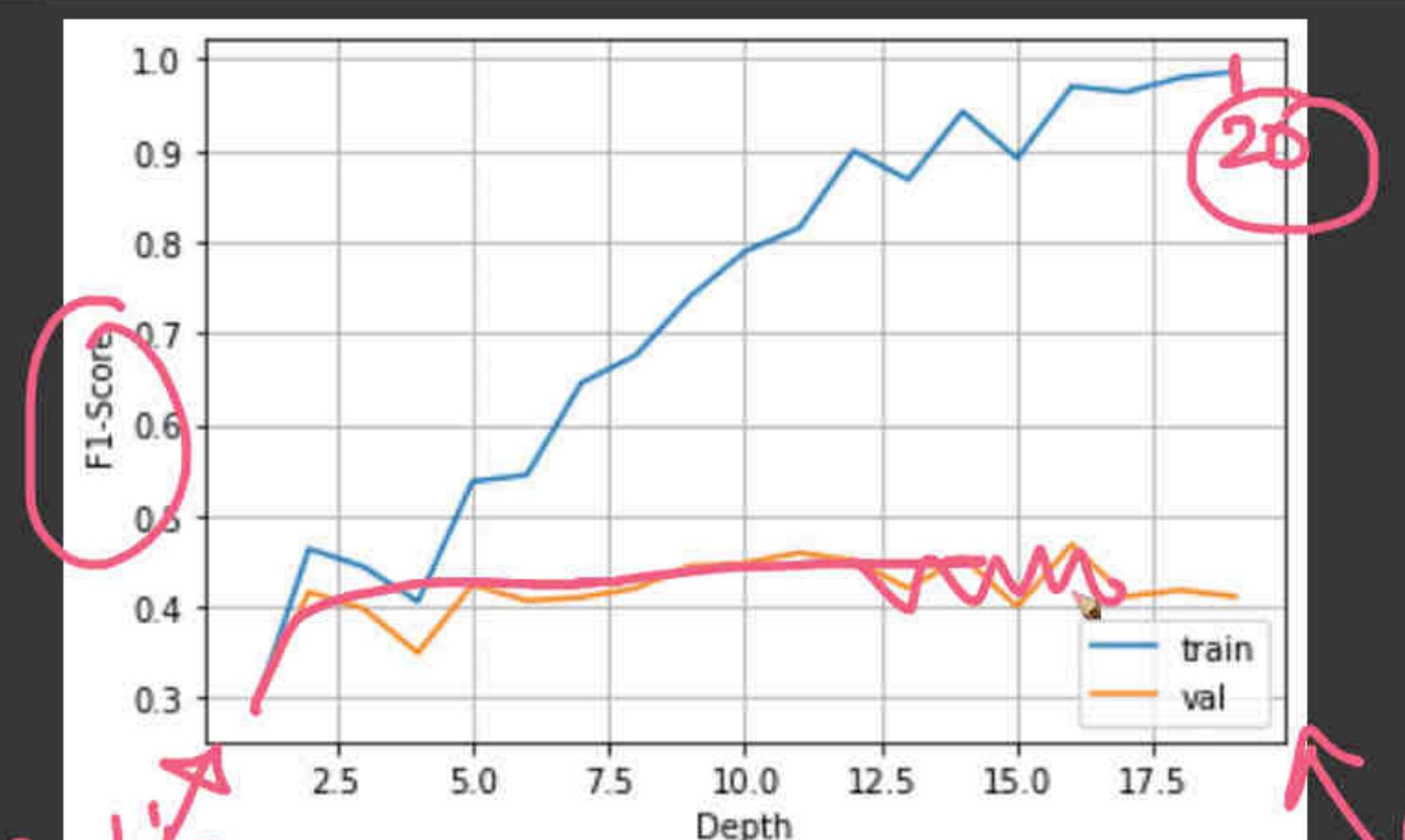
5 10.0 12.5 15.0
~~ma~~ Depth

```
[210] # Model with depth_best
      from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
best_idx = np.argmax(val_scores)
```

+ Code + Tex

RAM Disk

```
plt.plot(list(np.arange(1,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



800 fair pls 30 feet

under 17

Overfit

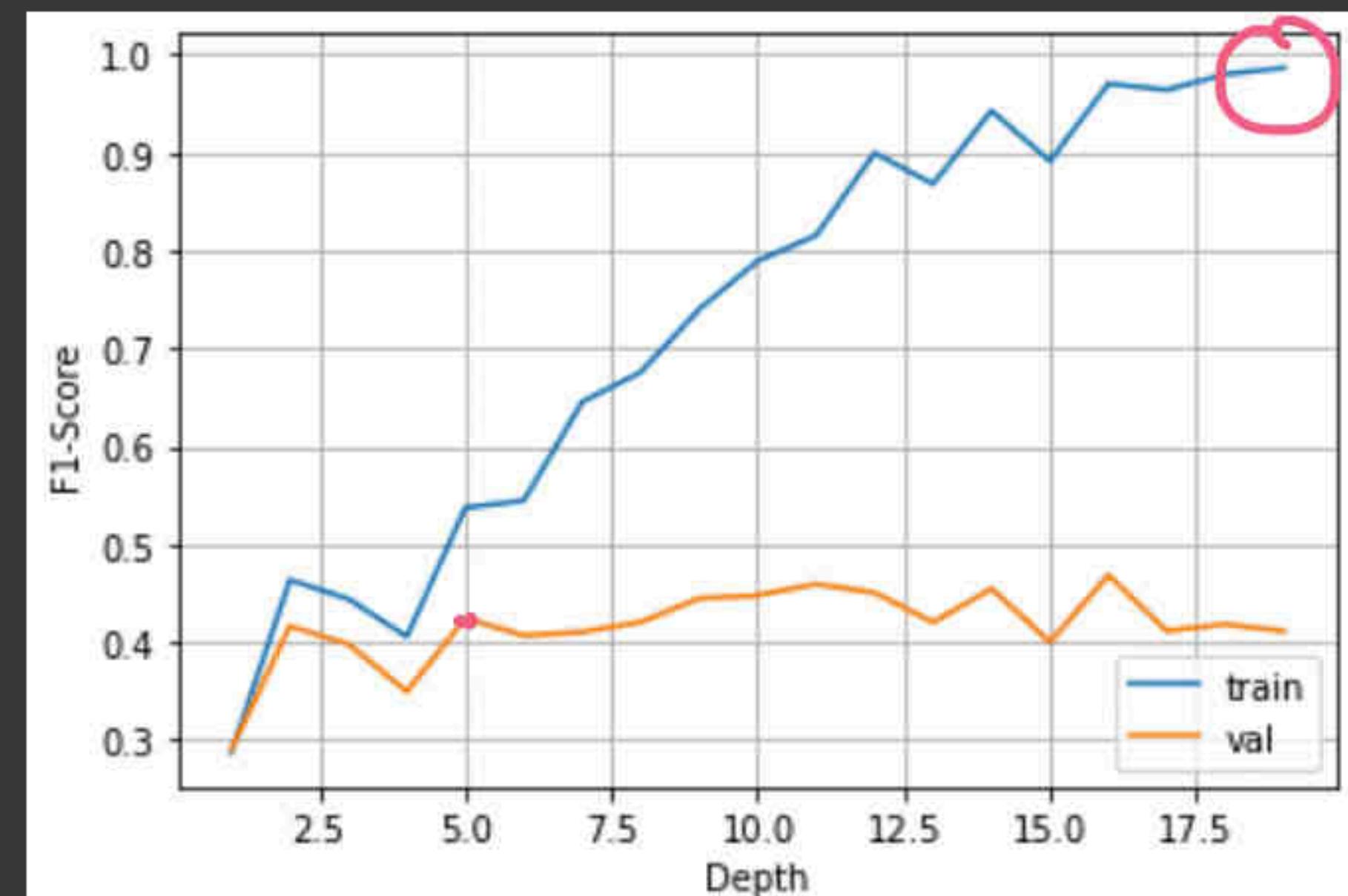
```
[✓] [210] # Model with depth_best
      from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
      best_idx = np.argmax(val_scores)
```

+ Code + Text

✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



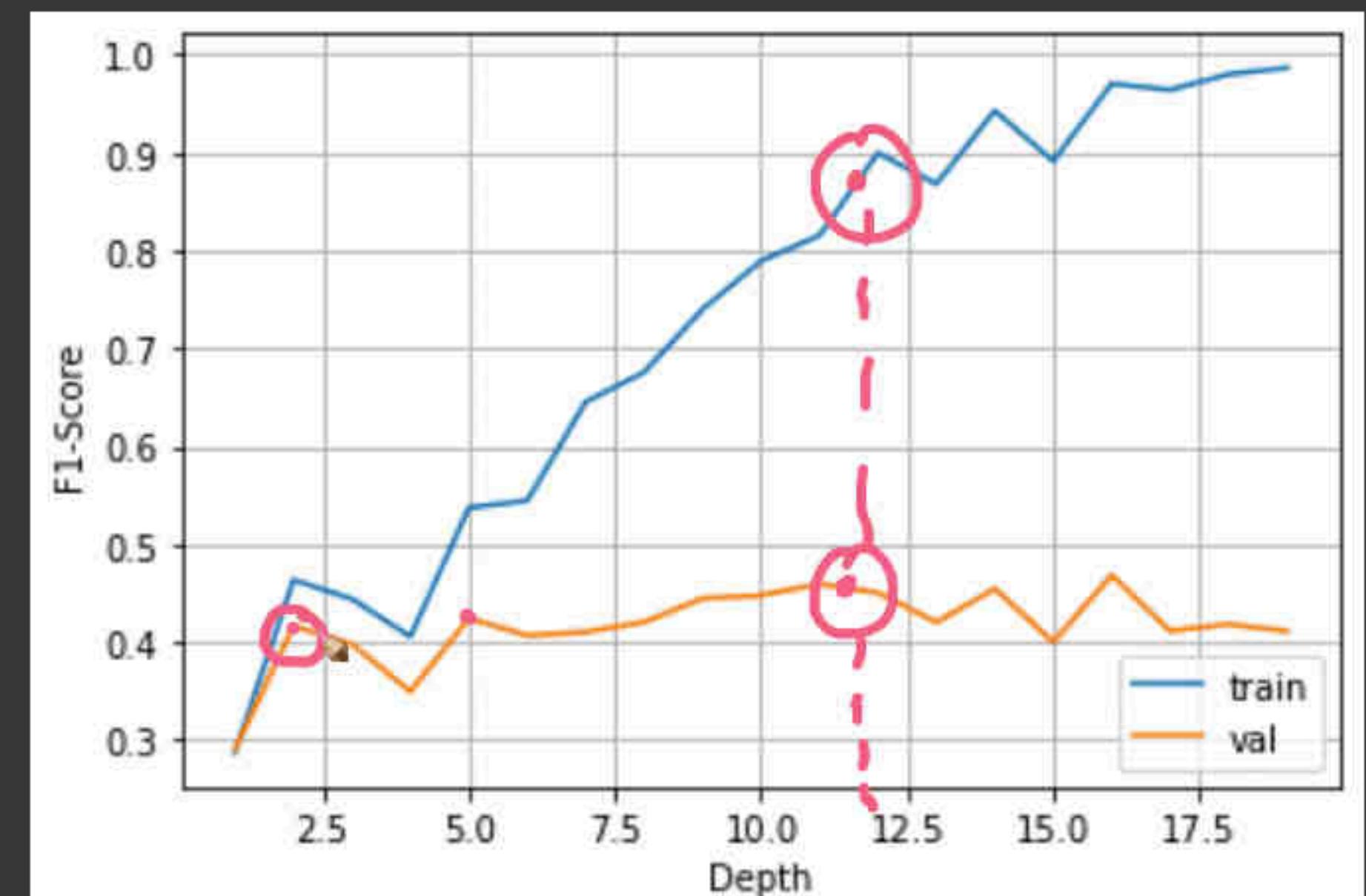
```
] # Model with depth_best
    from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
best_idx = np.argmax(val_scores)
```

+ Code + Text

✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```

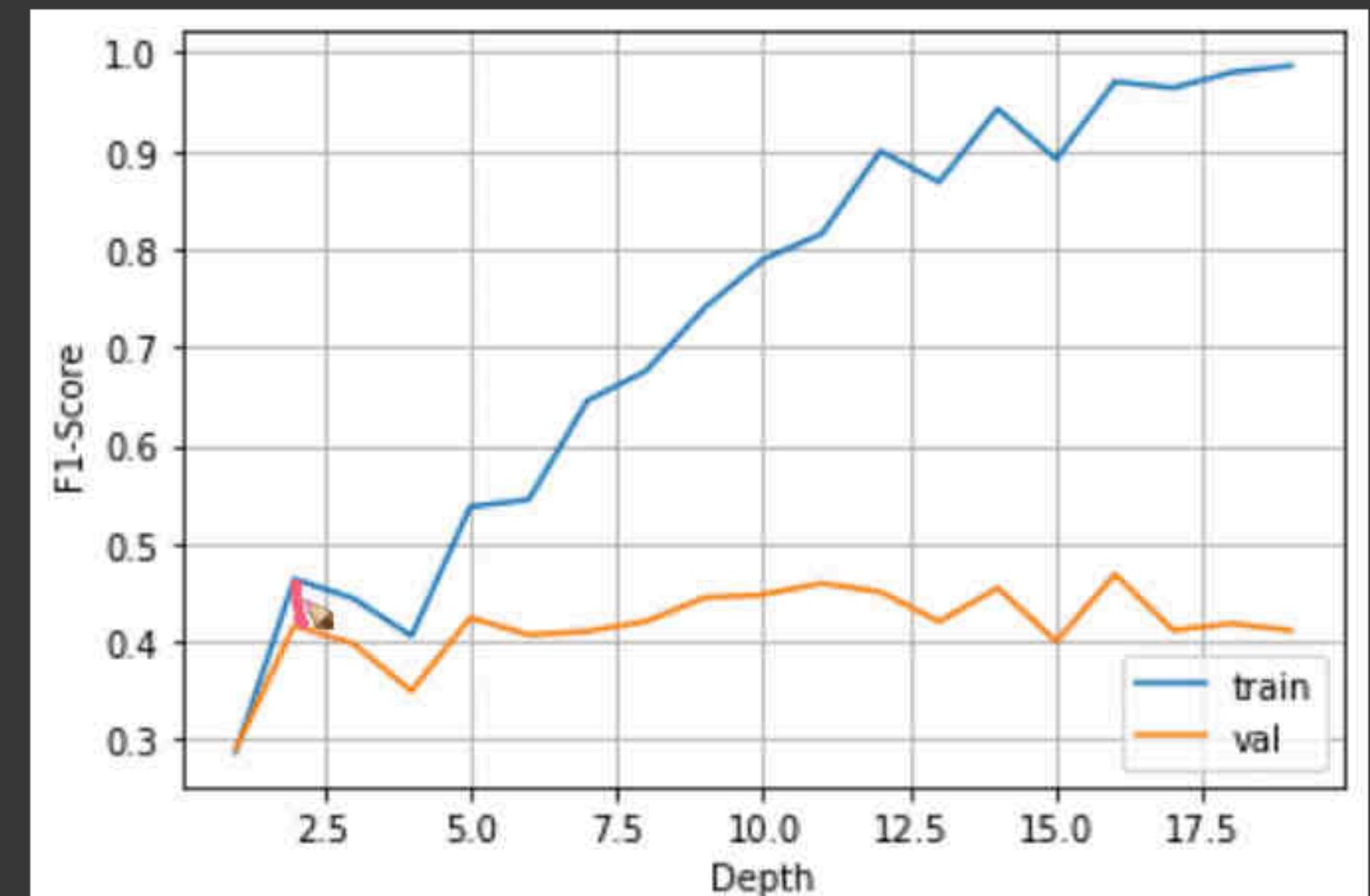


+ Code + Text

✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



[210] # Model with depth_best

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_recall_curve
```

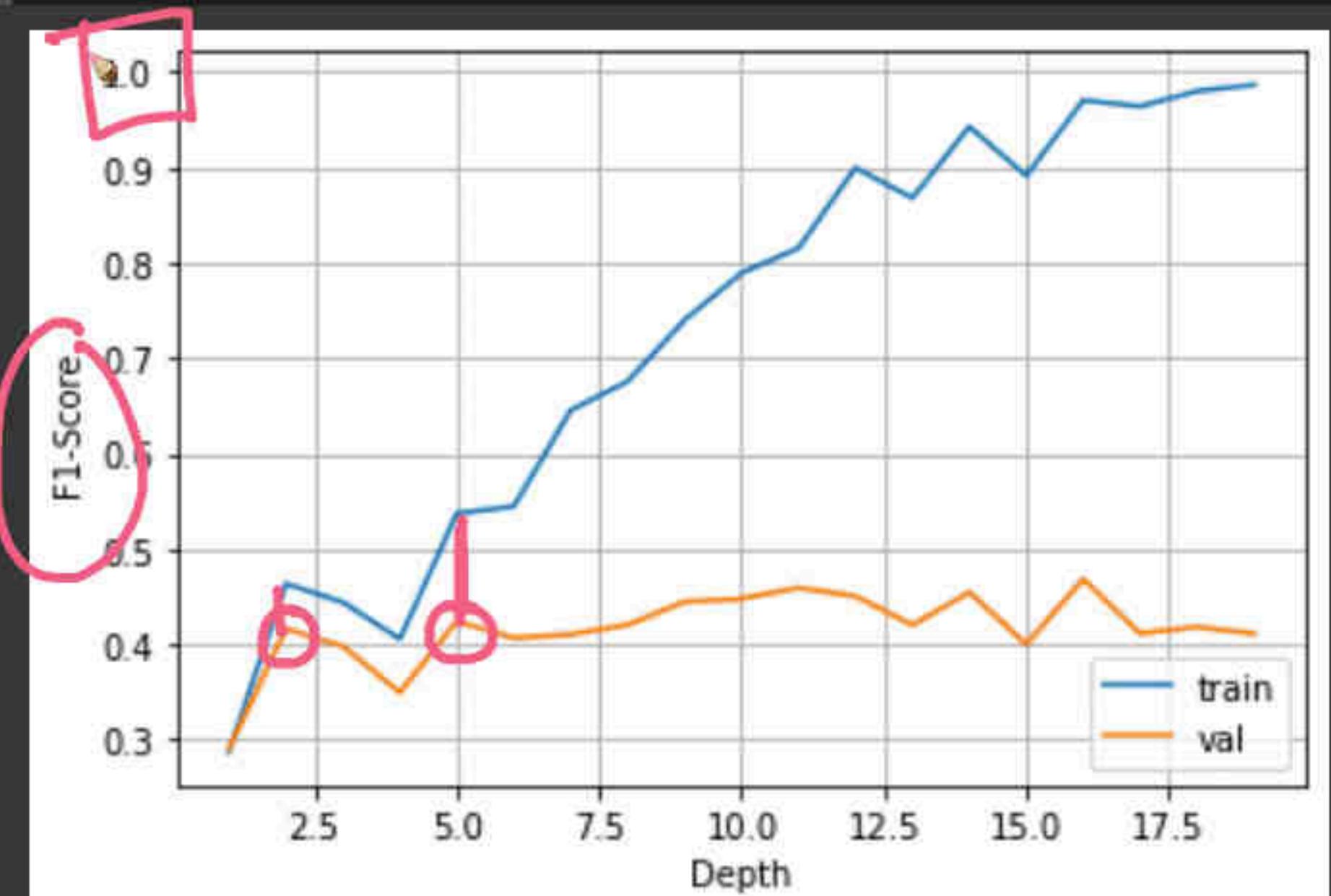
```
best_idx = np.argmax(val_scores)
```

+ Code + Text

✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```

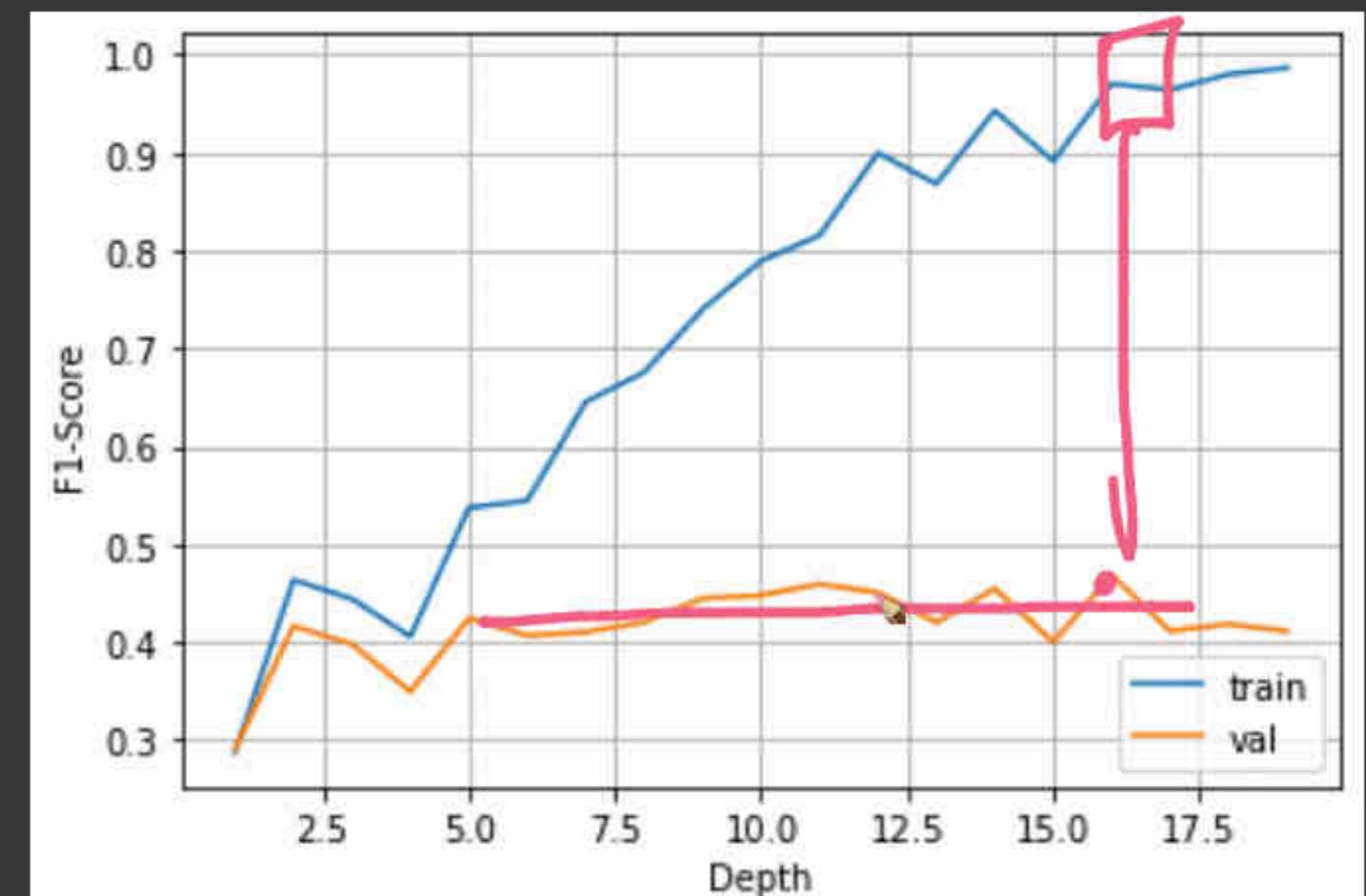


+ Code + Text

✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```

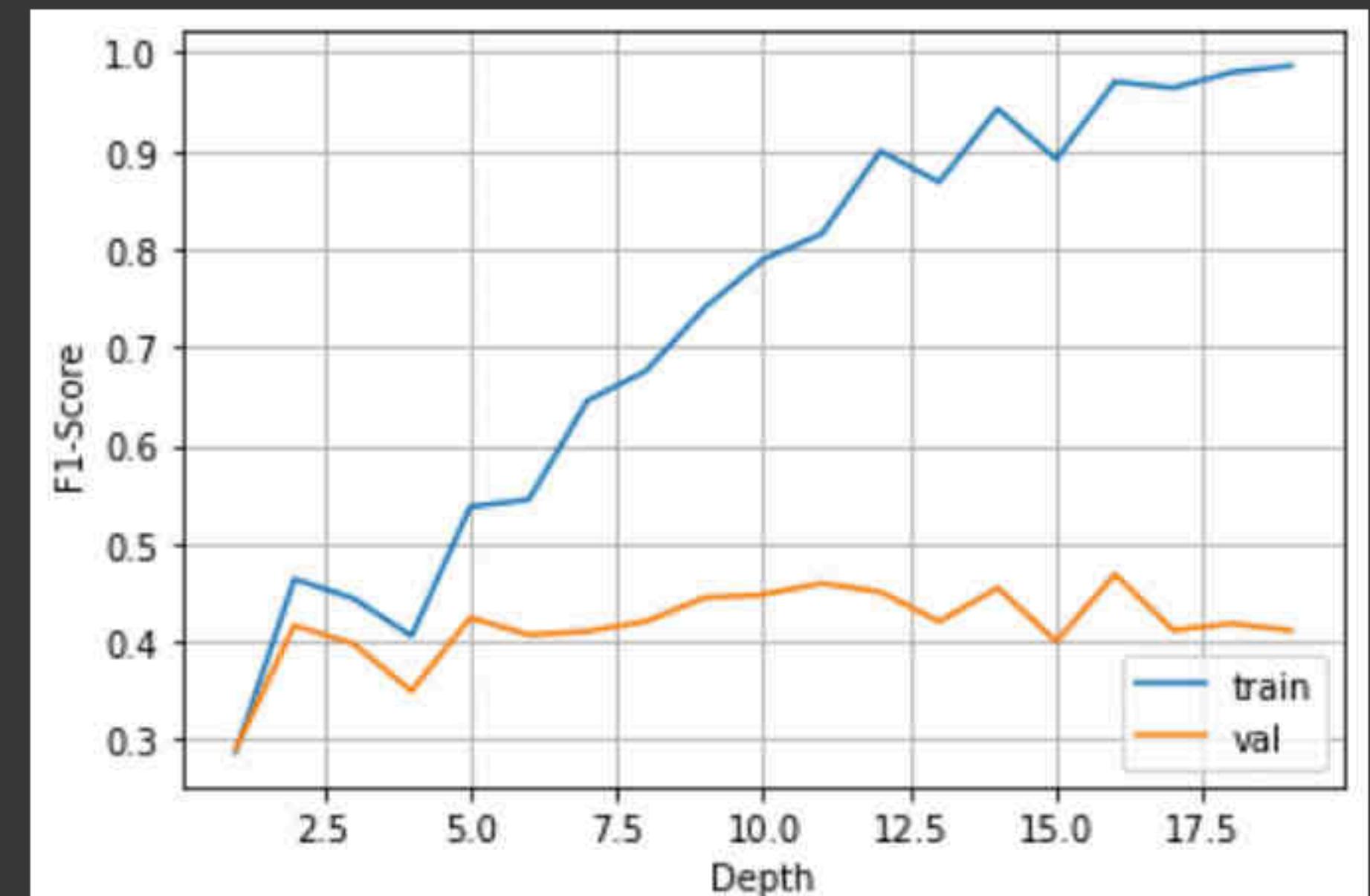


+ Code + Text

✓ RAM	
Disk	



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



[210] # Model with depth_best

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
```

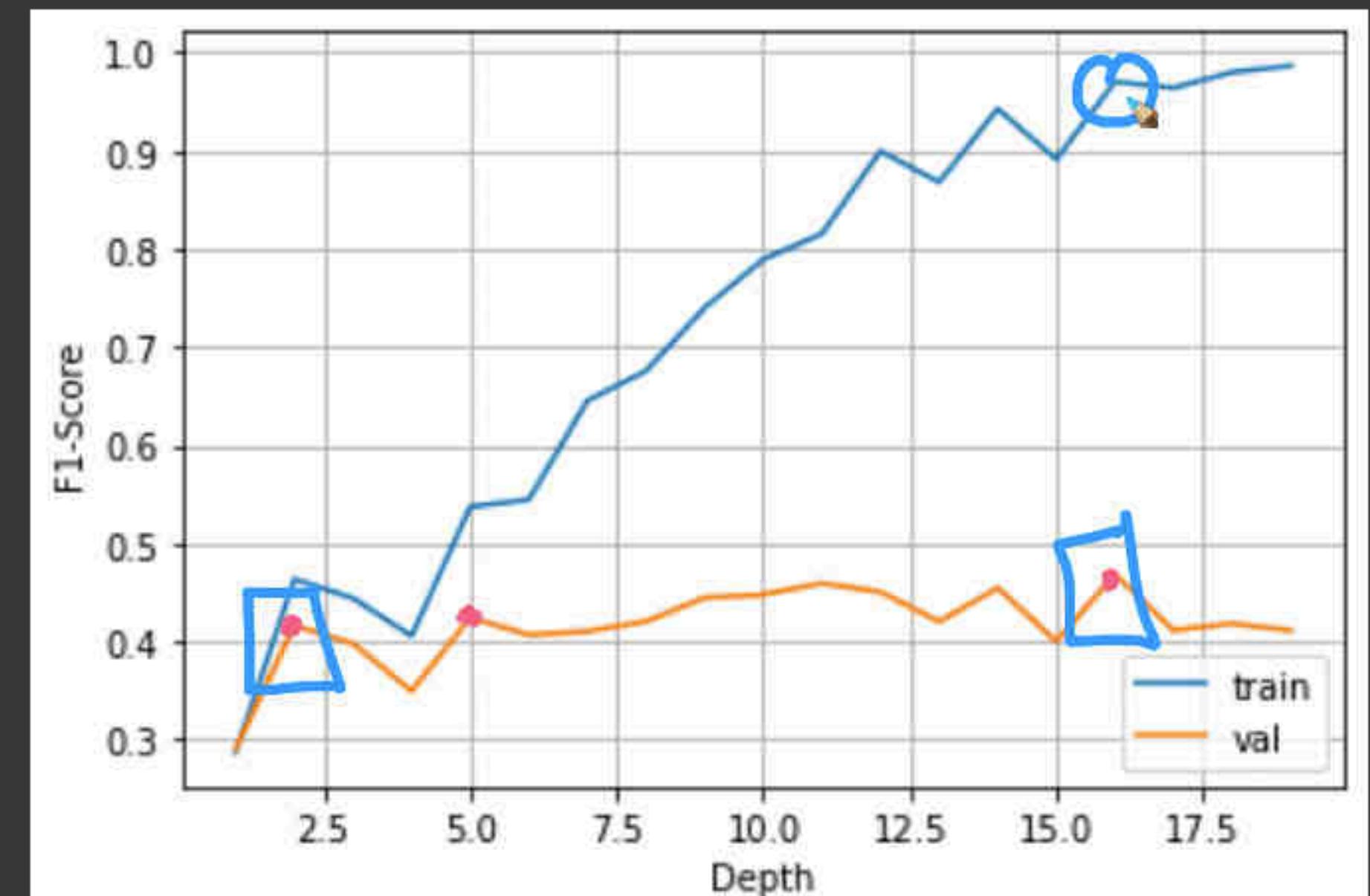
```
best_idx = np.argmax(val_scores)
```

+ Code + Text

✓ RAM Disk



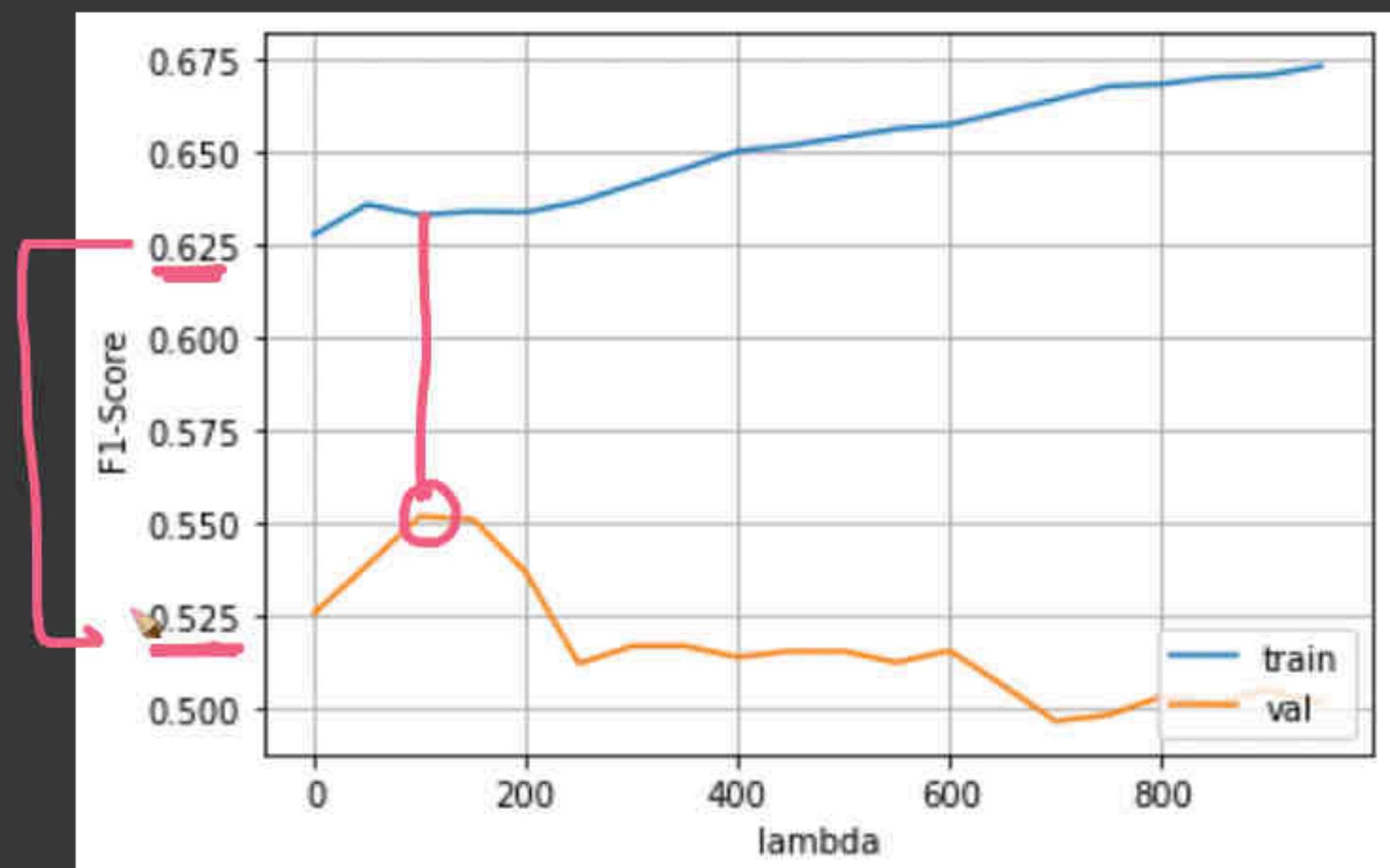
```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



```
[210] # Model with depth_best
      from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
best_idx = np.argmax(val_scores)
```

+ Code + Text

```
[ ] plt.grid()  
plt.show()
```



Logistic Reg

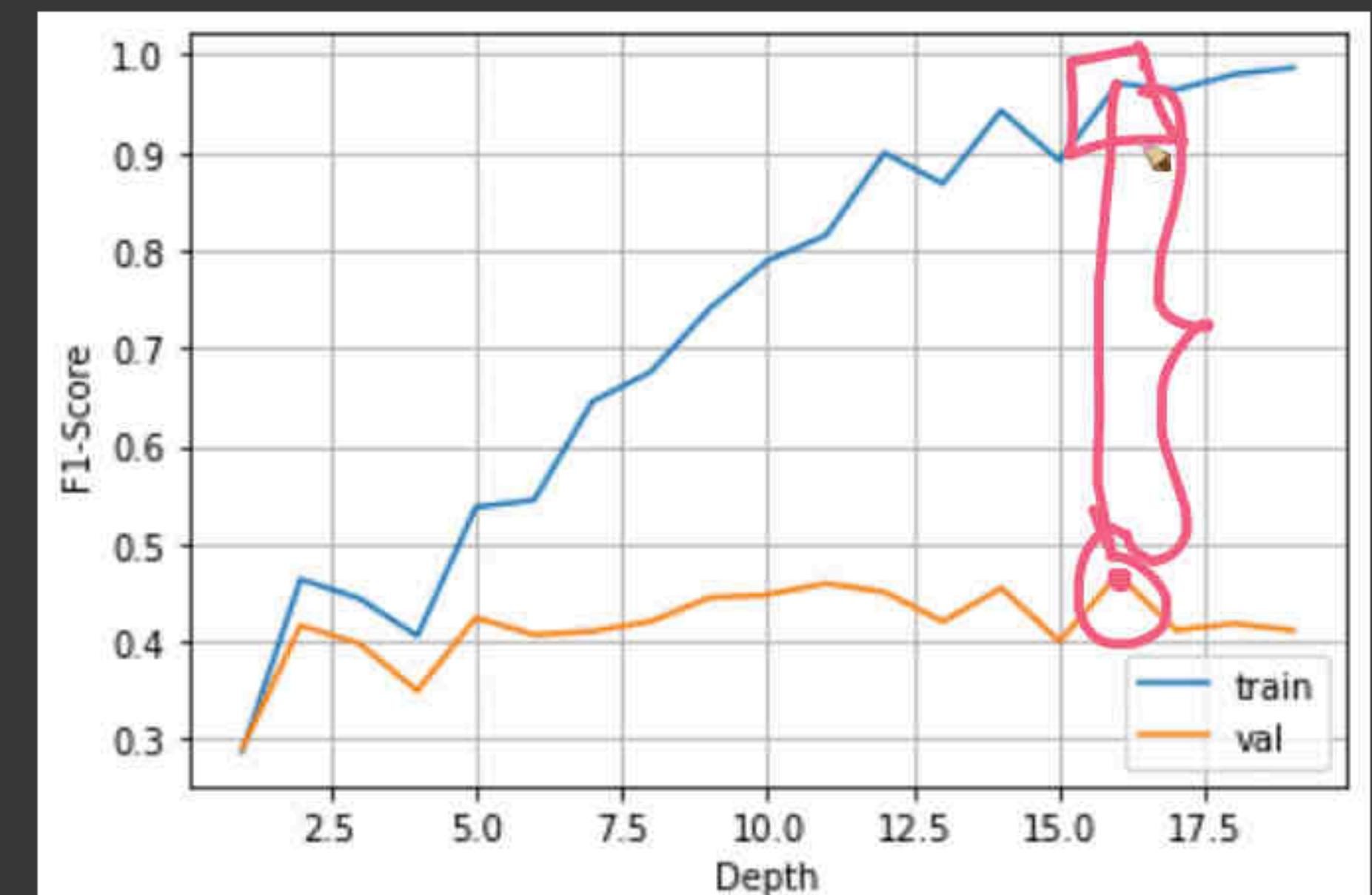
```
[ ] # minority class needs more weighting  
  
# Hyper-param tuning  
from sklearn.linear_model import LogisticRegression  
from sklearn.pipeline import make_pipeline  
from sklearn.metrics import f1_score
```

+ Code + Tex

✓	RAM	
✓	Disk	



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



$\{x\}$

11



∞ DT and RF.ipynb - Colaboratory | ∞ sklearn.tree.DecisionTreeClassif... | ∞ sklearn.ensemble.RandomFore... | ∞ sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | ∞ Imbalanced Data.ipynb - Colab +

colab.research.google.com/drive/1aJN0epwvGFkdUmxRBpBkxH6gykOnUagm

+ Code + Text

[] plt.grid()
plt.show()

{x}

File

Q

lambda

F1-Score

train

val

[] # minority class needs more weighting

[] # Hyper-param tuning

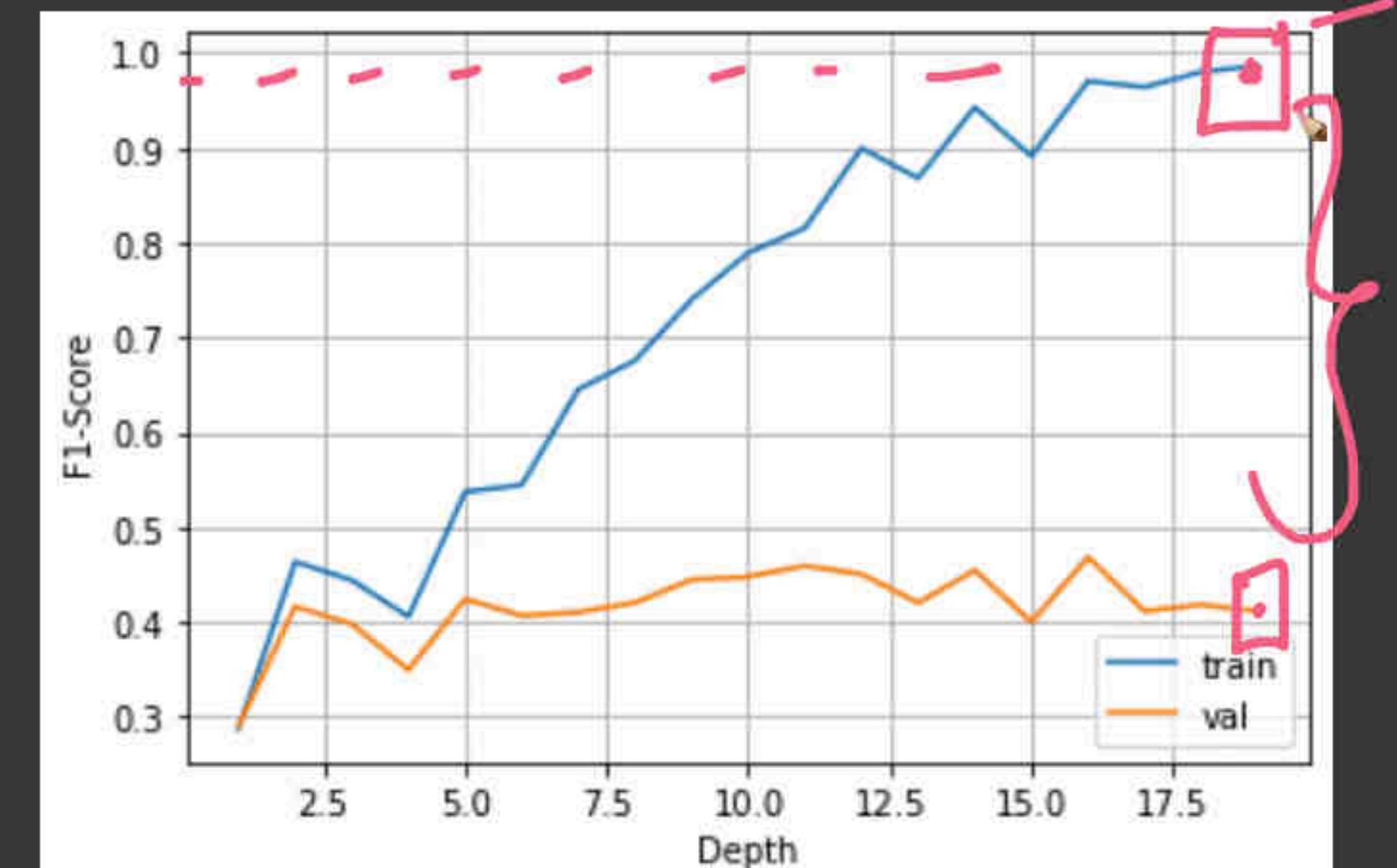
```
<> from sklearn.linear_model import LogisticRegression  
[ ] from sklearn.pipeline import make_pipeline  
[ ] from sklearn.metrics import f1_score
```

+ Code + Tex

✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



depth = 20

100

1

1

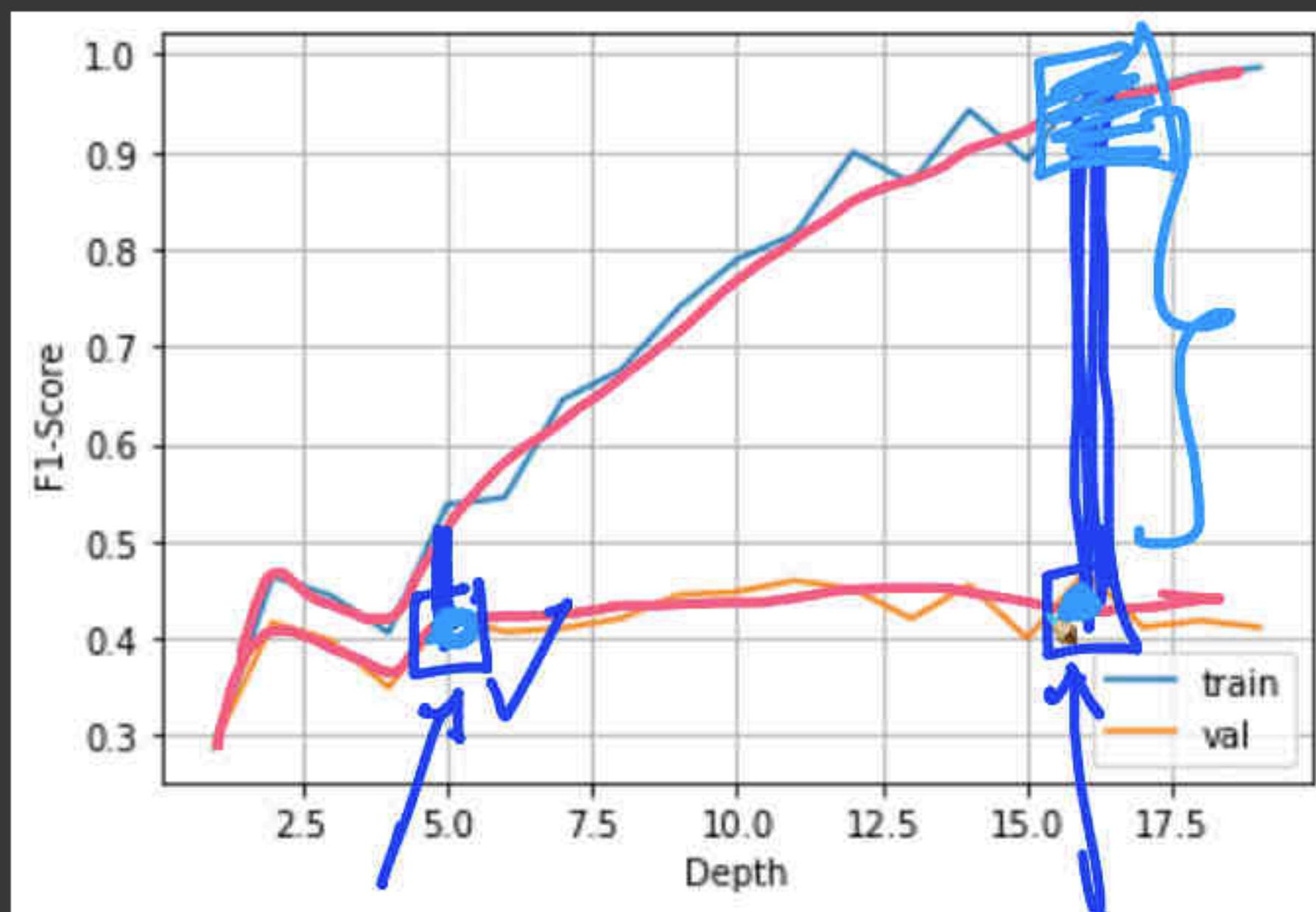
```
[210] # Model with depth_best
      from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
best_idx = np.argmax(val_scores)
```

+ Code + Tex

✓	RAM	<div style="width: 100px; height: 10px; background-color: #ccc; border: 1px solid black;"></div>
✓	Disk	<div style="width: 100px; height: 10px; background-color: #ccc; border: 1px solid black;"></div>



```
plt.plot(list(np.arange(1,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



800

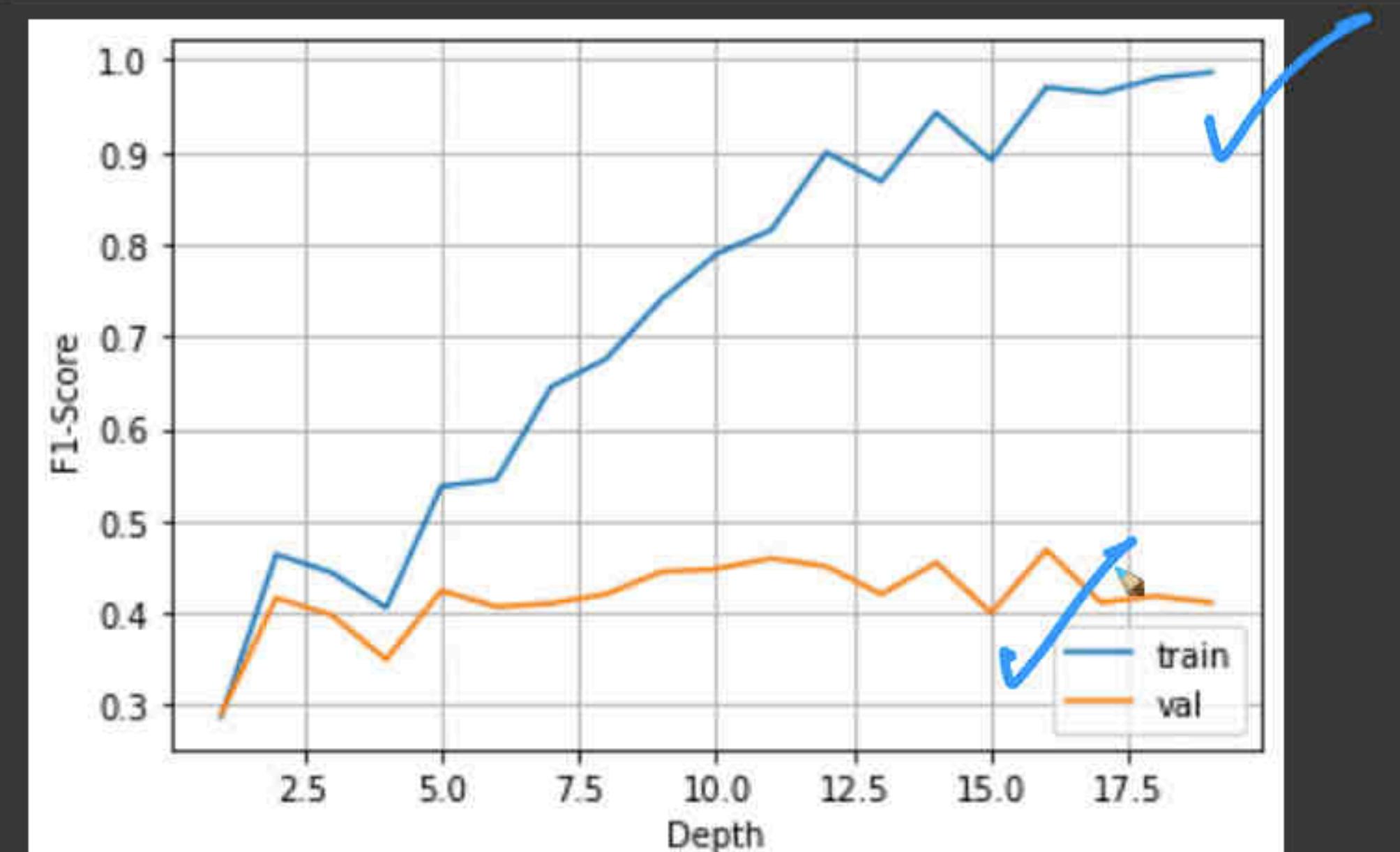
```
  ✓ [210] # Model with depth_best
          from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
>-
    best_idx = np.argmax(val_scores)
```

+ Code + Tex

- ✓ RAM Disk



```
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



✓ [210] # Model with depth_best

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_recall_curve
```

```
best_idx = np.argmax(val_scores)
```

 DT and RF.ipynb - Colaborator sklearn.tree.DecisionTreeClass

X | sklearn.ensemble.RandomFore X

sklearn.preprocessing.LabelEncoder

Target Encoder – Category En

∞ Imbalanced Data.ipynb – Colab

 Update : ...

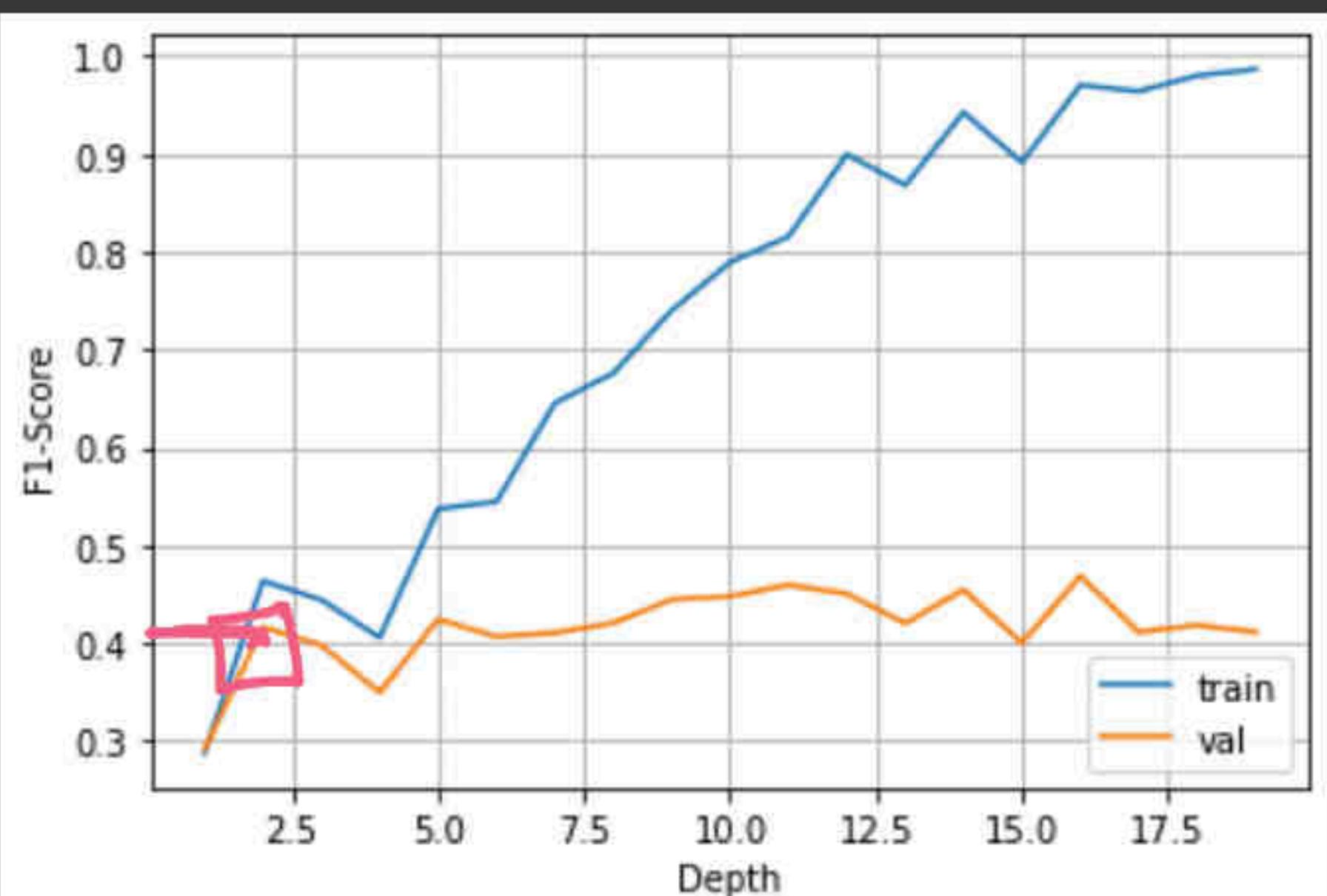
+ Code + Text

✓ RAM Disk

1

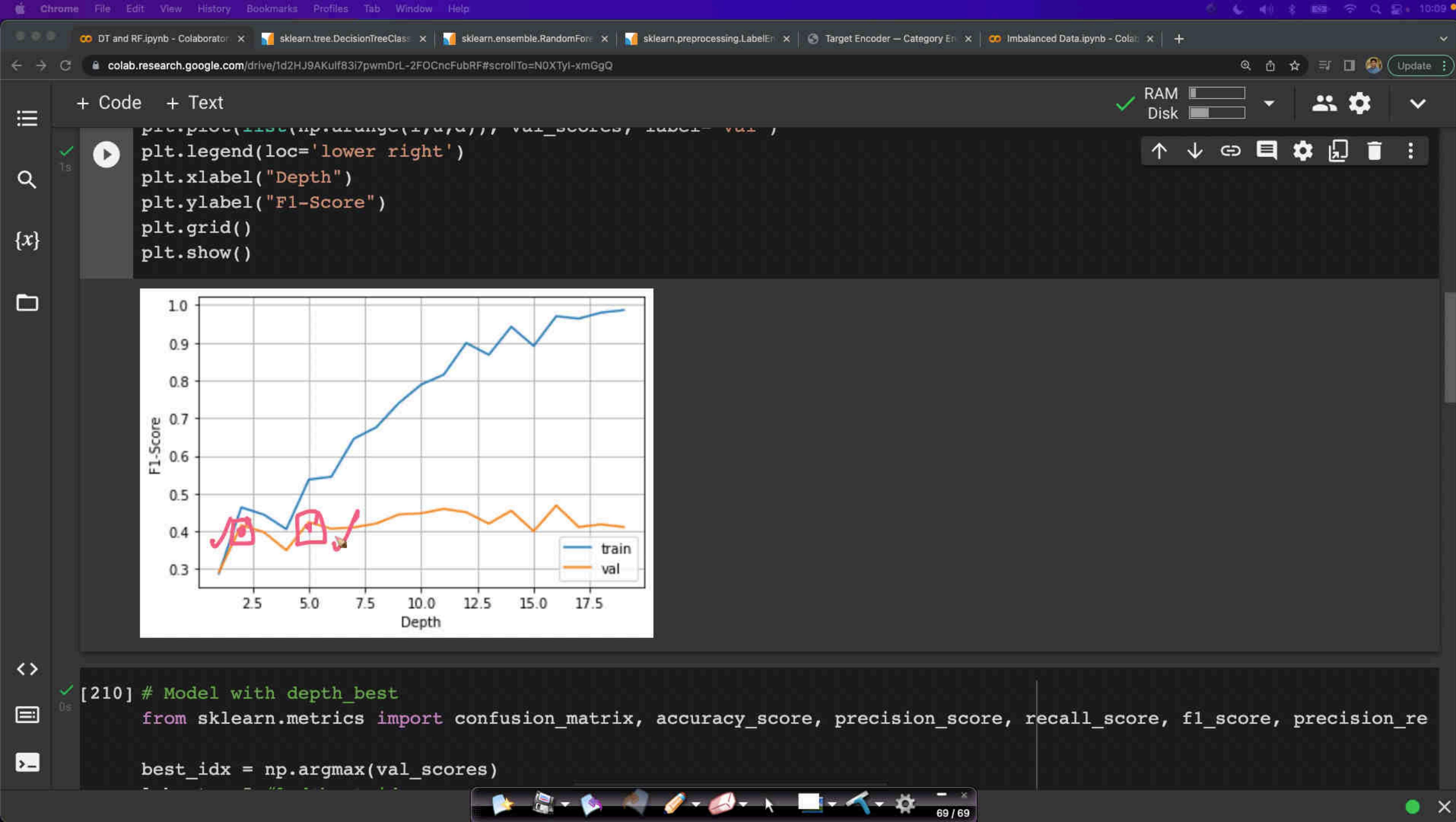
```
plt.legend(loc='lower right')
plt.xlabel("Depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```





A hand-drawn diagram of a stick figure smoking a cigarette, with a curly brace above it labeled "30 features". The figure is drawn with white lines on a black background. It has a circular head with a small circle for an eye, a simple body, and stick-like arms and legs. A curved line extends from its mouth to represent a cigarette, with a small circle at the end representing smoke. A pink curly brace is positioned above the figure, enclosing the head and upper body, with the text "30 features" written in pink.

```
[210] # Model with depth_best
      from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
best_idx = np.argmax(val_scores)
```



DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=xsY90Nb4rJg0

+ Code + Text

RAM Disk

Depth

{x}

Model with depth_best

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
```

best_idx = np.argmax(val_scores)

l_best = 5 #l+d*best_idx

print(l_best)

clf = DecisionTreeClassifier(random_state=0, max_depth=l_best, class_weight={ 0:0.1, 1:w })

clf.fit(X_train, y_train)

y_pred_val = clf.predict(X_val)

val_score = f1_score(y_val, y_pred_val)

print(val_score)

confusion_matrix(y_val, y_pred_val)

5

0.42424242424242425

array([[164, 80],
 [15, 35]])

[211] # Feature importance

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=xsY90Nb4rJg0

+ Code + Text

RAM Disk

RAM Disk

Code

Text

Q

{x}

D

best_idx = np.argmax(val_scores)
l_best = 2 #l+d*best_idx
print(l_best)
clf = DecisionTreeClassifier(random_state=0, max_depth=l_best, class_weight={ 0:0.1, 1:w })
clf.fit(X_train, y_train)

y_pred_val = clf.predict(X_val)
val_score = f1_score(y_val, y_pred_val)

print(val_score)

confusion_matrix(y_val, y_pred_val)

2
0.41618497109826585
array([[157, 87],
 [14, 36]])

[211] # Feature importance

importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title

RAM Disk

Pred

actual

0 1

0 1

FP

164 80

15 35

1 ←

71 / 73

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab... | +

scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

feature_ 1/4

scikit-learn

Prev Up Next

scikit-learn 1.1.0

Other versions

Please cite us if you use the software.

sklearn.tree.DecisionTreeClassifier

Examples using sklearn.tree.DecisionTreeClassif...

clf

ccp_alpha : non-negative float, default=0.0

Complexity parameter used for Minimal Cost-Complexity Pruning. The subtree with the largest cost complexity that is smaller than `ccp_alpha` will be chosen. By default, no pruning is performed. See [Minimal Cost-Complexity Pruning](#) for details.

New in version 0.22.

Attributes:

classes_ : ndarray of shape (n_classes,) or list of ndarray

The classes labels (single output problem), or a list of arrays of class labels (multi-output problem).

feature_importances_ : ndarray of shape (n_features,)

Return the feature importances.

max_features_ : int

The inferred value of `max_features`.

n_classes_ : int or list of int

The number of classes (for single output problems), or a list containing the number of classes for each output (for multi-output problems).

n_features_ : int

DEPRECATED: The attribute `n_features_` is deprecated in 1.0 and will be removed in

Weighting for each feature



Toggle Menu

∞ DT and RF.ipynb - Colaboratory | ∞ sklearn.tree.DecisionTreeClass... | ∞ sklearn.ensemble.RandomFore... | ∞ sklearn.preprocessing.LabelEnc... | ∞ Target Encoder — Category Enc... | ∞ Imbalanced Data.ipynb - Colab... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=xsY90Nb4rJg0

+ Code + Text

RAM Disk

array([[157, 87],
 [14, 36]])

{x} ✓ 0s

▶ # Feature importance
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot

Feature Importance

0.25
0.20
0.15

74 / 76

∞ DT and RF.ipynb - Colaboratory | ∞ sklearn.tree.DecisionTreeClassif... | ∞ sklearn.ensemble.RandomFore... | ∞ sklearn.preprocessing.LabelEnc... | ∞ Target Encoder — Category Enc... | ∞ Imbalanced Data.ipynb - Colab... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=xsY90Nb4rJg0

+ Code + Text

RAM Disk

array([[157, 87],
 [14, 36]])

{x} ✓ 0s

▶ # Feature importance
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot

Feature Importance

0.25
0.20
0.15

< >

75 / 77

 DT and RF.ipynb - Colaborator | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEr... | Target Encoder — CategoryEn... | Imbalanced Data.ipynb - Colab

- Code + Text

RAM Disk

1

```
array([[157,  87],  
       [ 14,  36]])
```

$f_1 f_2 \dots f_d$

0.24 0.24 ~ ~

```
# Feature importance
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot
```



∞ DT and RF.ipynb - Colaboratory | ∞ sklearn.tree.DecisionTreeClassif... | ∞ sklearn.ensemble.RandomFore... | ∞ sklearn.preprocessing.LabelEnc... | ∞ Target Encoder — Category Enc... | ∞ Imbalanced Data.ipynb - Colab... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=xsY90Nb4rJg0

+ Code + Text

RAM Disk

array([[157, 87],
 [14, 36]])

{x} ✓ 0s

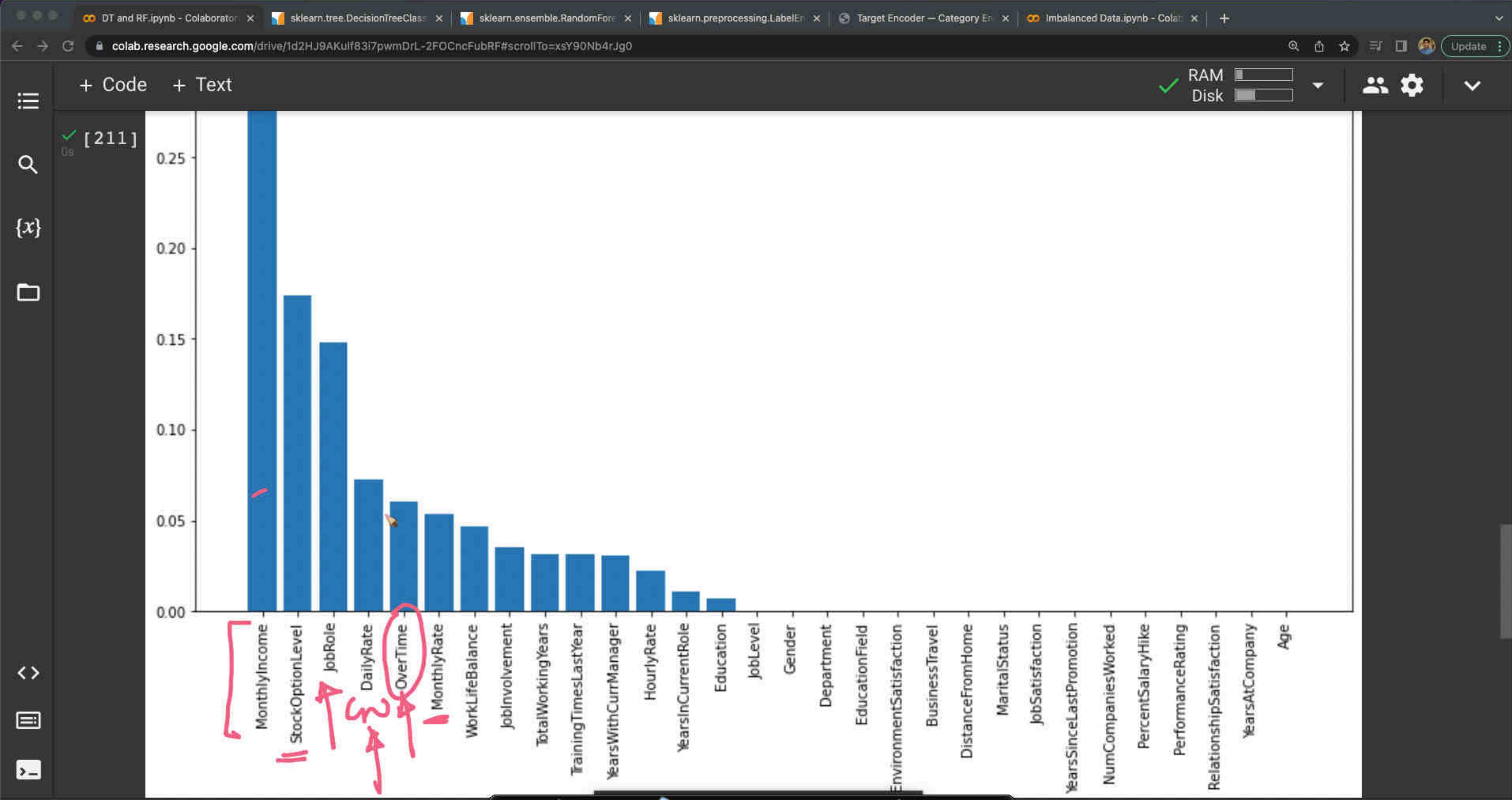
▶ # Feature importance
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance

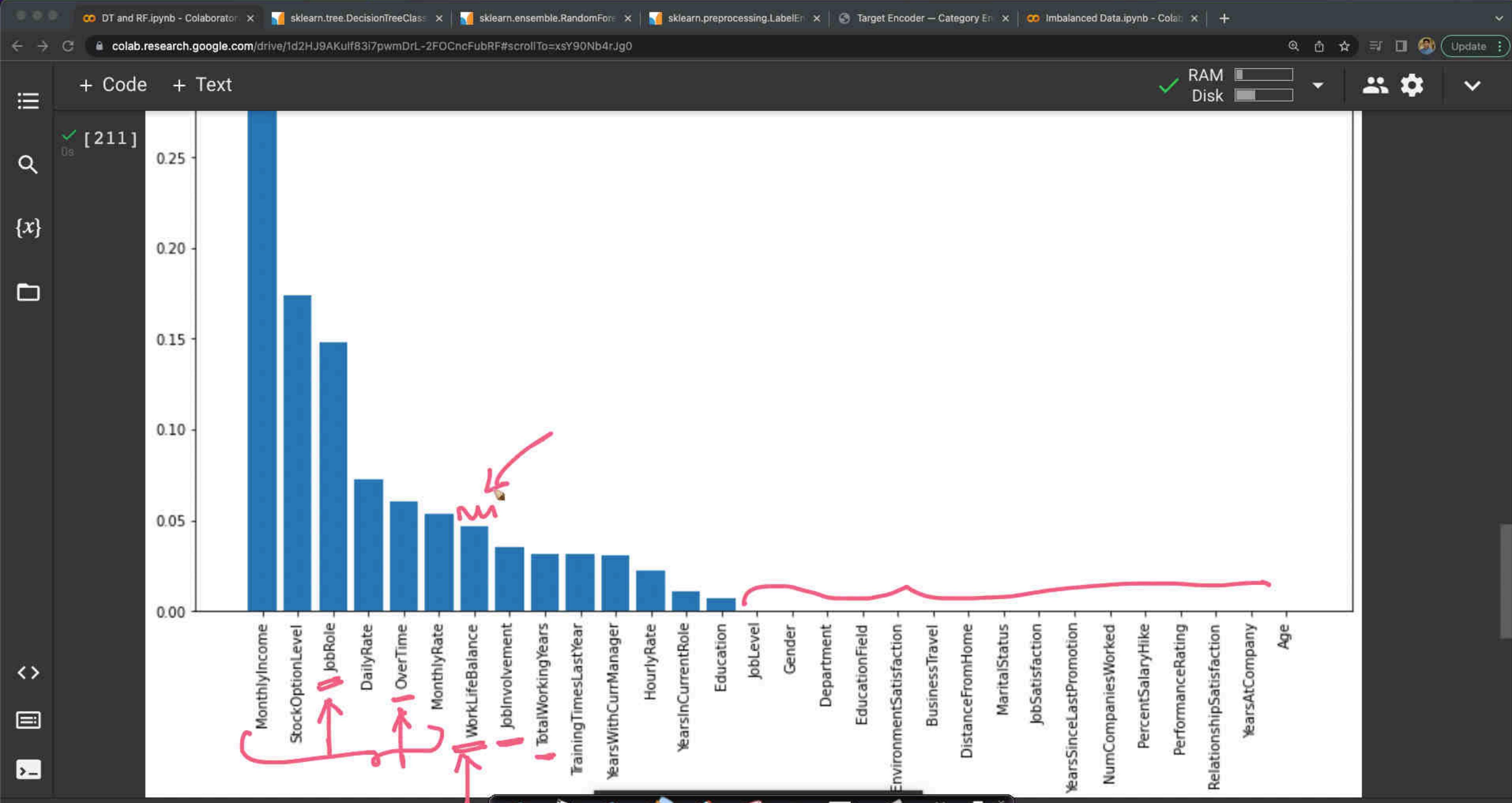
{ plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot

Feature Importance

Feature	Importance
Feature 1	~0.27
Feature 2	~0.17
Feature 3	~0.15

77 / 79





DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=xI_3aNxqqkuu

+ Code + Text

RAM Disk

[216] train_scores = []
val_scores = []

{x}
l=1
u=20
d=1
w=3.0 ←
num_learners=100 ←
row_sampling_rate = 0.75 ←

for depth in np.arange(l,u,d):
 clf = RandomForestClassifier(max_depth=depth, max_samples=row_sampling_rate, n_estimators=num_learners, random_st
 clf.fit(X_train, y_train)
 train_y_pred = clf.predict(X_train)
 val_y_pred = clf.predict(X_val)
 train_score = f1_score(y_train, train_y_pred)
 val_score = f1_score(y_val, val_y_pred)
 train_scores.append(train_score)
 val_scores.append(val_score)

M I

0.1 3.0

0.1 1.0

[217] import matplotlib.pyplot as plt
plt.figure()

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=xI_3aNxqqkuu

+ Code + Text

RAM Disk ✓

[216] train_scores = []
val_scores = []

l=1
u=20
d=1
w=3.0
num_learners=100
row_sampling_rate = 0.75

$\frac{d'}{d}$

$D_{T\gamma}^n \rightarrow S_{l,M}^{0.75} M_I$

$\frac{M}{N} = 0.75$

for depth in np.arange(l,u,d):
 clf = RandomForestClassifier(max_depth=depth, max_samples=row_sampling_rate, n_estimators=num_learners, random_st
 clf.fit(X_train, y_train)
 train_y_pred = clf.predict(X_train)
 val_y_pred = clf.predict(X_val)
 train_score = f1_score(y_train, train_y_pred)
 val_score = f1_score(y_val, val_y_pred)
 train_scores.append(train_score)
 val_scores.append(val_score)

[217] import matplotlib.pyplot as plt
plt.figure()

81 / 83

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder — Category Encoders | Imbalanced Data.ipynb - Colaboratory | +

- Code + Text

✓ RAM Disk

1

```
[216] train_scores = []
      val_scores = []

l=1
u=20
d=1
w=3.0
num_learners=100
row_sampling_rate = 0.75
```

Xgboost

```
for depth in np.arange(1,u,d):
    clf = RandomForestClassifier(max_depth=depth, max_samples=row_sampling_rate, n_estimators=num_learners, random_st
    clf.fit(X_train, y_train)
    train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val_score)
```

```
[217] import matplotlib.pyplot as plt  
plt.figure()
```

+ Code + Text

RAM Disk

```
[216] train_scores = []
      val_scores = []

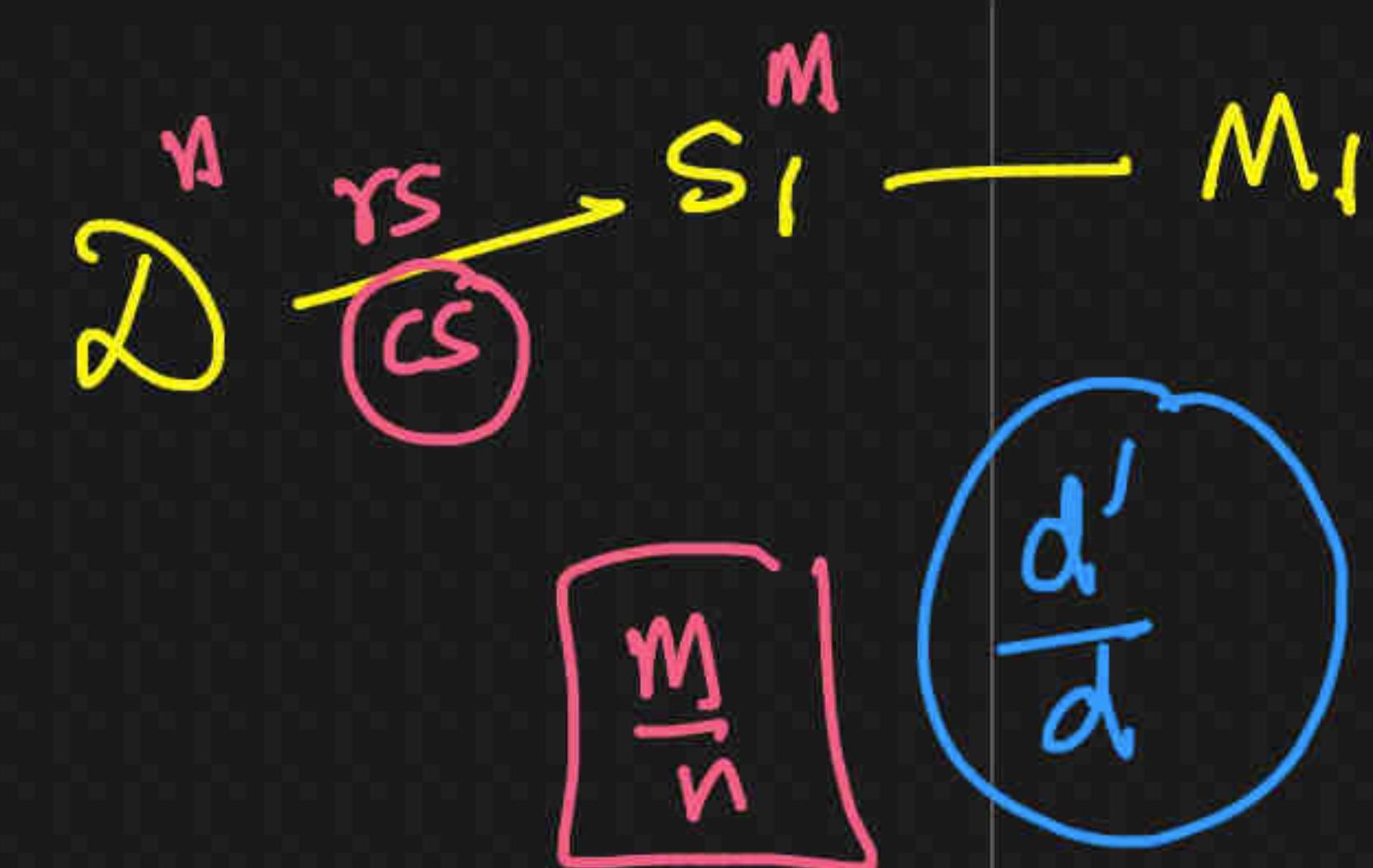
l=1
u=20
d=1
w=3.0
num_learners=100
row_sampling_rate = 0.75
```

```
for depth in np.arange(1,u,d):
    clf = RandomForestClassifier(max_depth=depth, max_samples=row_sampling_rate, n_estimators=num_learners, random_st
    clf.fit(X_train, y_train)
    train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val_score)
```

11

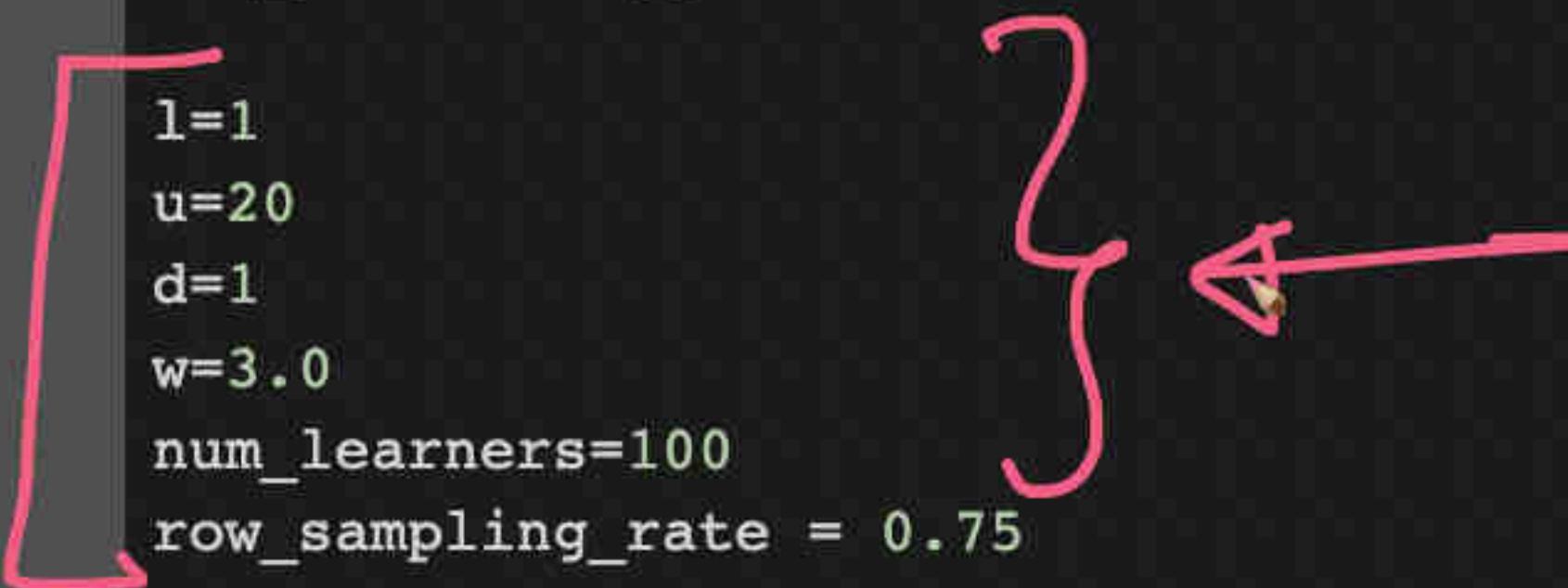
[217] import matplotlib.pyplot as plt

```
plt.figure()
```



+ Code + Text

✓ RAM Disk



```
for depth in np.arange(1,u,d):
    clf = RandomForestClassifier(max_depth=depth, max_samples=row_sampling_rate, n_estimators=num_learners, random_st
    clf.fit(X_train, y_train)
    train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val_score)
```

```
[217] import matplotlib.pyplot as plt  
plt.figure()
```

```
+ Code + Text
```

The screenshot shows a Jupyter Notebook interface. On the left, there are two code cells. The top cell contains the following code:

```
train_scores = []
val_scores = []

l=1
u=20
d=1
w=3.0
num_learners=100
row_sampling_rate = 0.75
```

The bottom cell has a play button icon and is currently empty. To the right of the cells is a toolbar with various icons for navigating and managing the notebook.

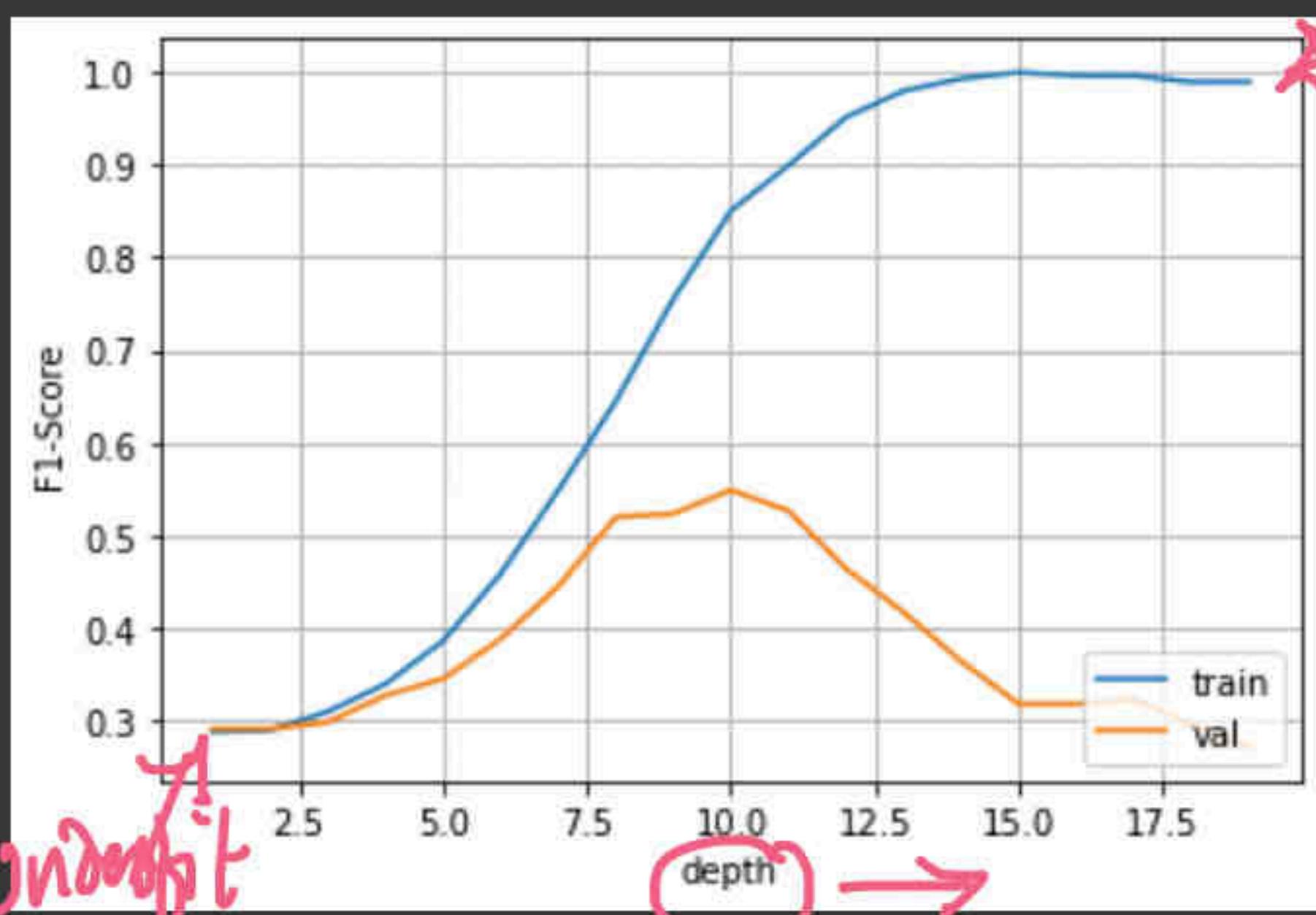
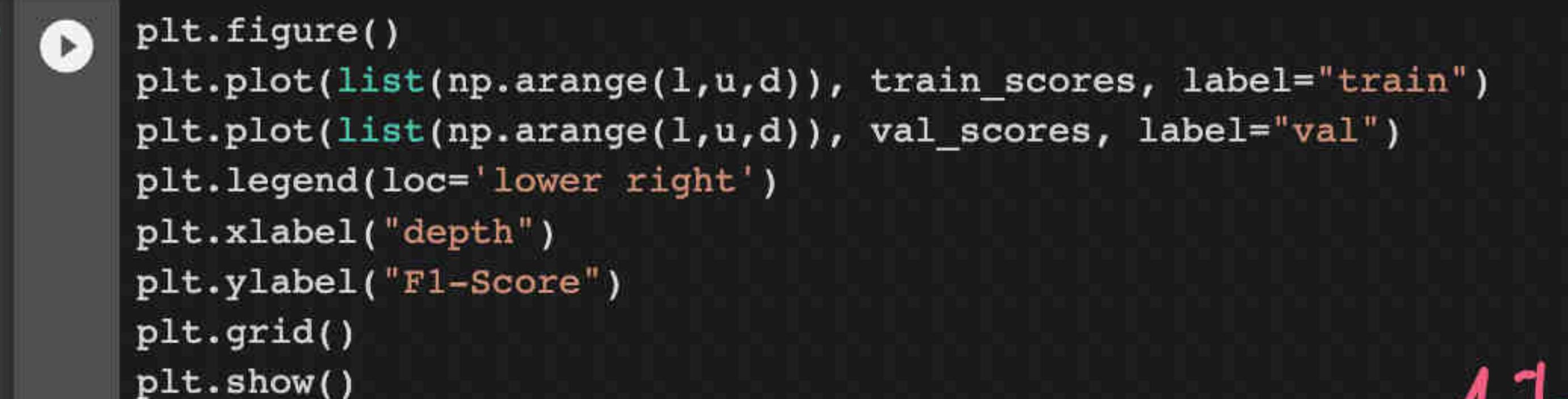
```
for depth in np.arange(1,u,d):
    clf = RandomForestClassifier(max_depth=depth, max_samples=row_sampling_rate, n_estimators=num_learners, random_st
    clf.fit(X_train, y_train)
    train_y_pred = clf.predict(X_train)
    val_y_pred = clf.predict(X_val)
    train_score = f1_score(y_train, train_y_pred)
    val_score = f1_score(y_val, val_y_pred)
    train_scores.append(train_score)
    val_scores.append(val_score)
```

✓ [217] import matplotlib.pyplot as plt

```
plt.figure()
```

+ Code + Text

RAM Disk



— overfit

depth ↑
m ↓

1

#baap
ter

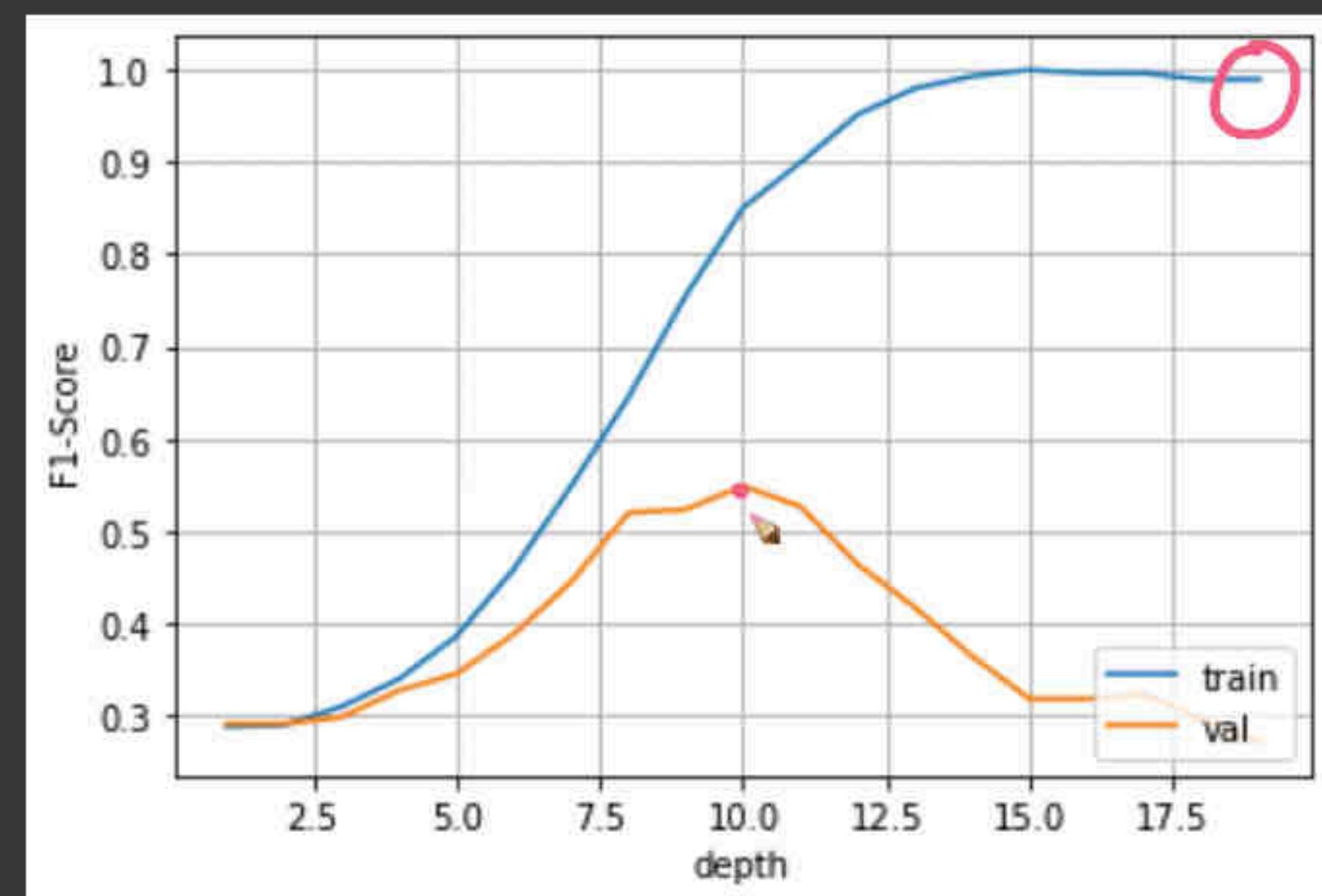
✓ [218] # Model with depth best

+ Code + Tex

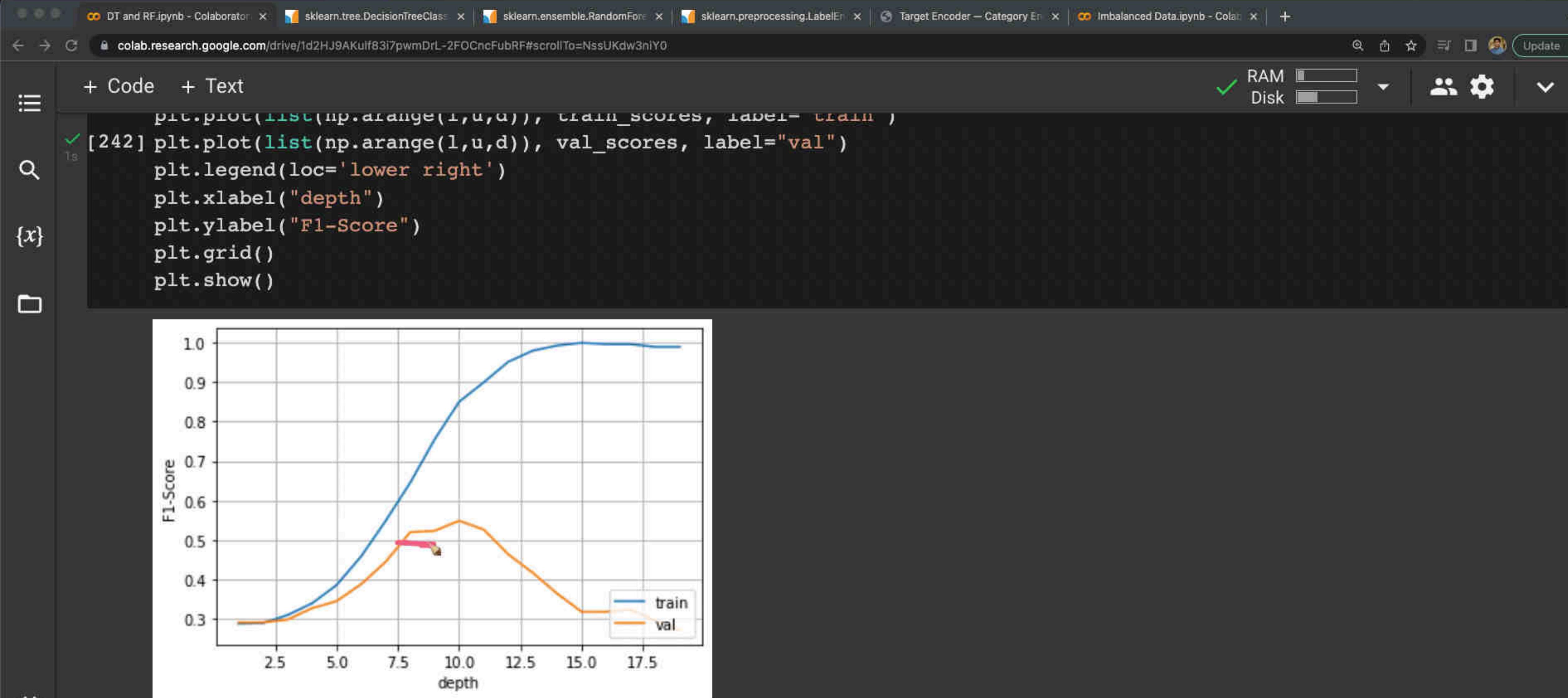
- ✓ RAM Disk



```
plt.figure()
plt.plot(list(np.arange(l,u,d)), train_scores, label="train")
plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()
```



✓ [218] # Model with depth best



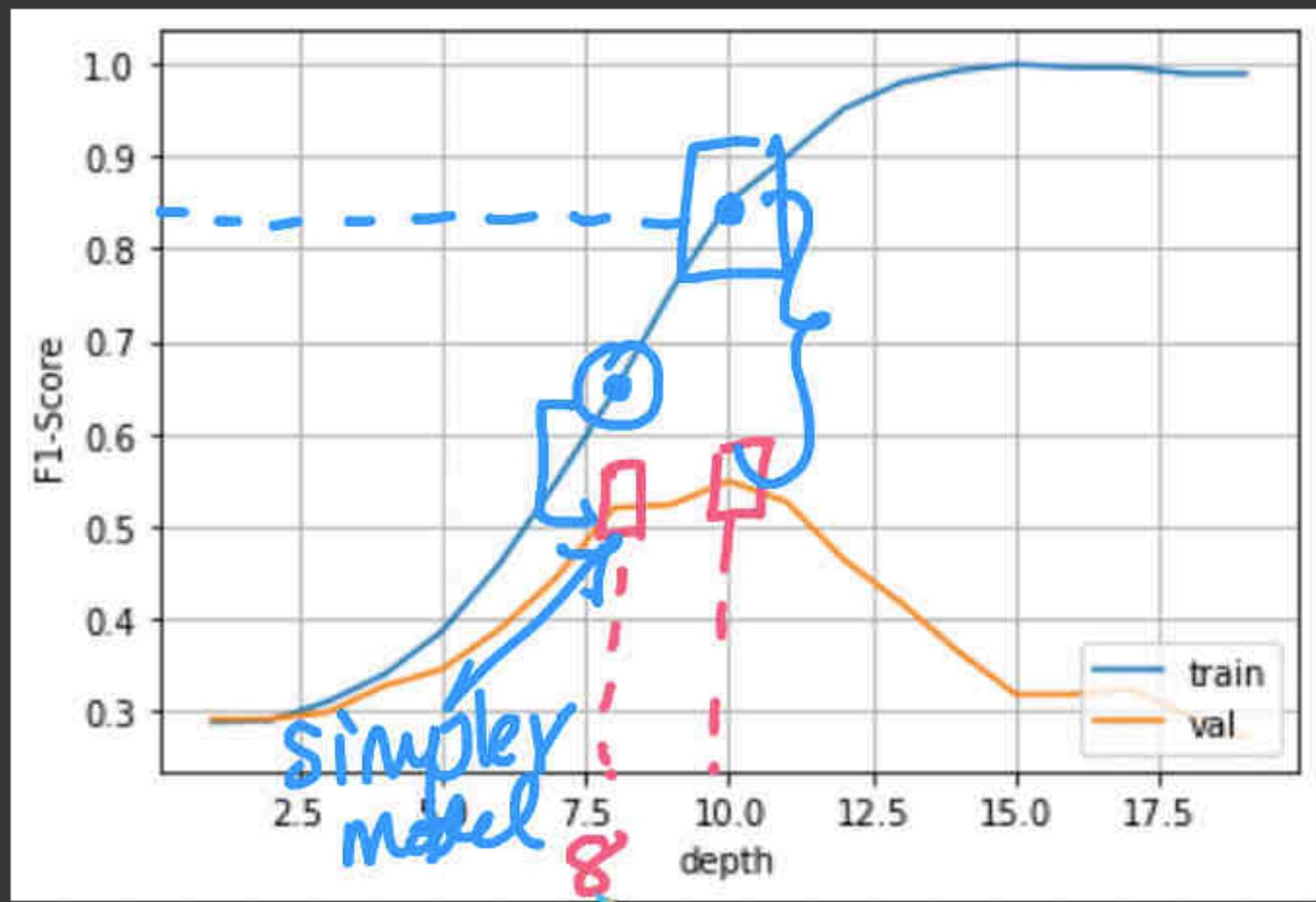
[218] # Model with depth_best
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category En... | Imbalanced Data.ipynb - Colab... | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=NssUKdw3niYO

+ Code + Text

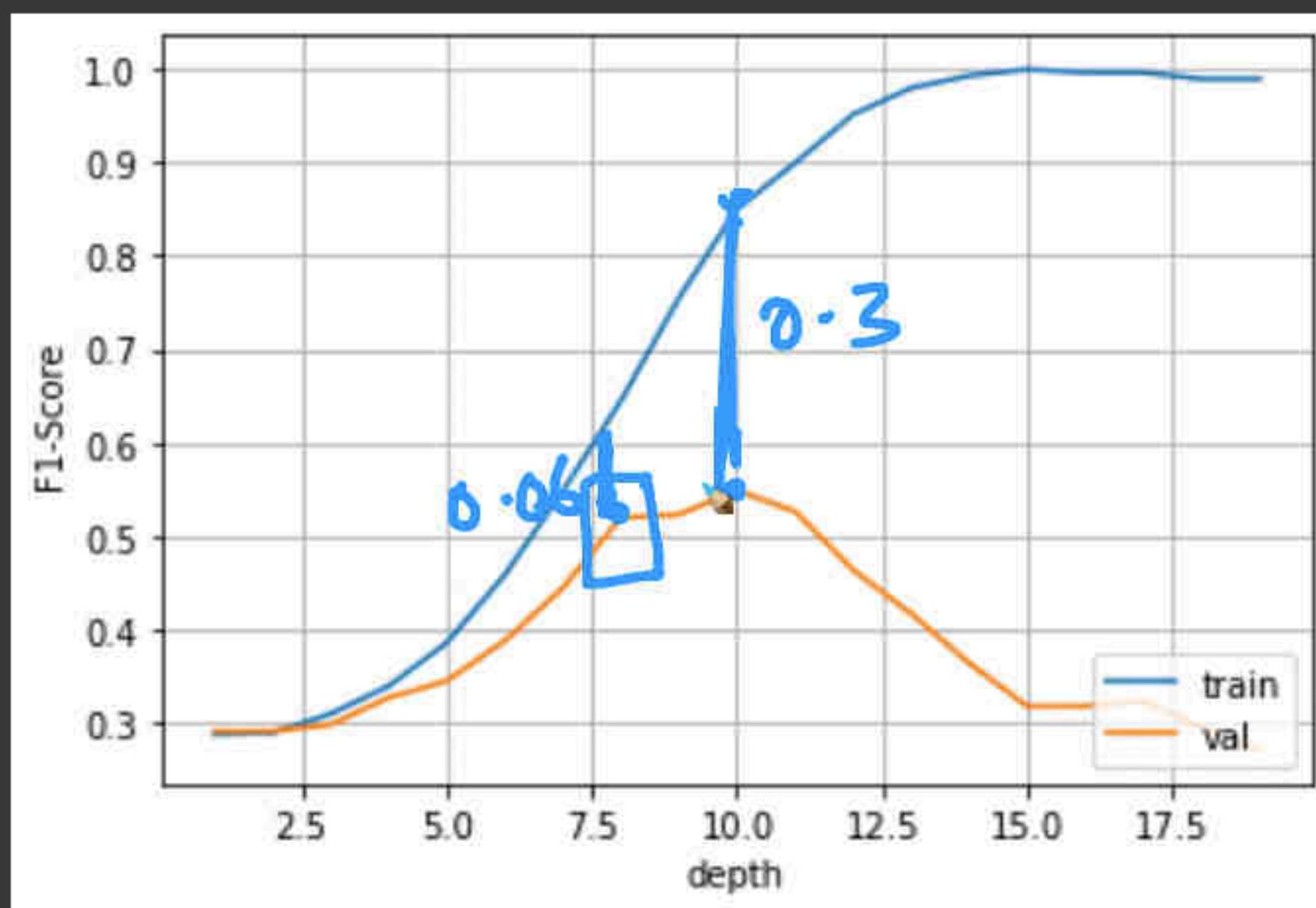
[242] plt.plot(list(np.arange(l,u,d)), val_scores, label="val")
plt.legend(loc='lower right')
plt.xlabel("depth")
plt.ylabel("F1-Score")
plt.grid()
plt.show()



[218] # Model with depth_best
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re

+ Code + Text

```
plt.plot(list(np.arange(1,u,u)), train_scores, label= "train")
[242] plt.plot(list(np.arange(1,u,d)), val_scores, label="val")
    plt.legend(loc='lower right')
    plt.xlabel("depth")
    plt.ylabel("F1-Score")
    plt.grid()
    plt.show()
```



```
[218] # Model with depth_best
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_re
```

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClassifer | sklearn.ensemble.RandomForestClassifier | sklearn.preprocessing.LabelEncoder | Target Encoder - Category Encoding | Imbalanced Data.ipynb - Colaboratory | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=QX-DZt3bsmqh

+ Code + Text

RAM Disk

depth

Model with depth_best

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, precision_recall_curve
```

best_idx = np.argmax(val_scores)

l_best = 8 #l+d*best_idx

```
clf = RandomForestClassifier(max_depth=l_best, max_samples = row_sampling_rate, n_estimators=num_learners, random_state=42)
```

clf.fit(X_train, y_train)

y_pred_val = clf.predict(X_val)

val_score = f1_score(y_val, y_pred_val)

print(val_score)

confusion_matrix(y_val, y_pred_val)

0.5194805194805195

array([[180, 64],
 [10, 40]])

<>

[219] # Feature importance

```
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
```

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=QX-DZt3bsmgh

+ Code + Text

[239] `print(val_score)`

`confusion_matrix(y_val, y_pred_val)`

2
0.41618497109826585
array([[157, 87],
 [14, 36]])

[211] # Feature importance

```
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot
```

Feature Importance

0.25

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=QX-DZt3bsmgh

+ Code + Text

RAM Disk

Code

clf = RandomForestClassifier(max_depth=l_best, max_samples = row_sampling_rate, n_estimators=100)

clf.fit(X_train, y_train)

y_pred_val = clf.predict(X_val)

val_score = f1_score(y_val, y_pred_val)

print(val_score)

confusion_matrix(y_val, y_pred_val)

0.5194805194805195

array([[180, 64],
 [10, 40]])

[219] # Feature importance

```
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot
```

Feature Importance

93 / 95

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=QX-DZt3bsmqh

+ Code + Text

RAM Disk

[239]

```
y_pred_val = clf.predict(X_val)
val_score = f1_score(y_val, y_pred_val)

print(val_score)

confusion_matrix(y_val, y_pred_val)
```

2
0.41618497109826585
array([[157, 87],
 [14, 36]])

DT: 87
14

RF: 64
10

[211] # Feature importance

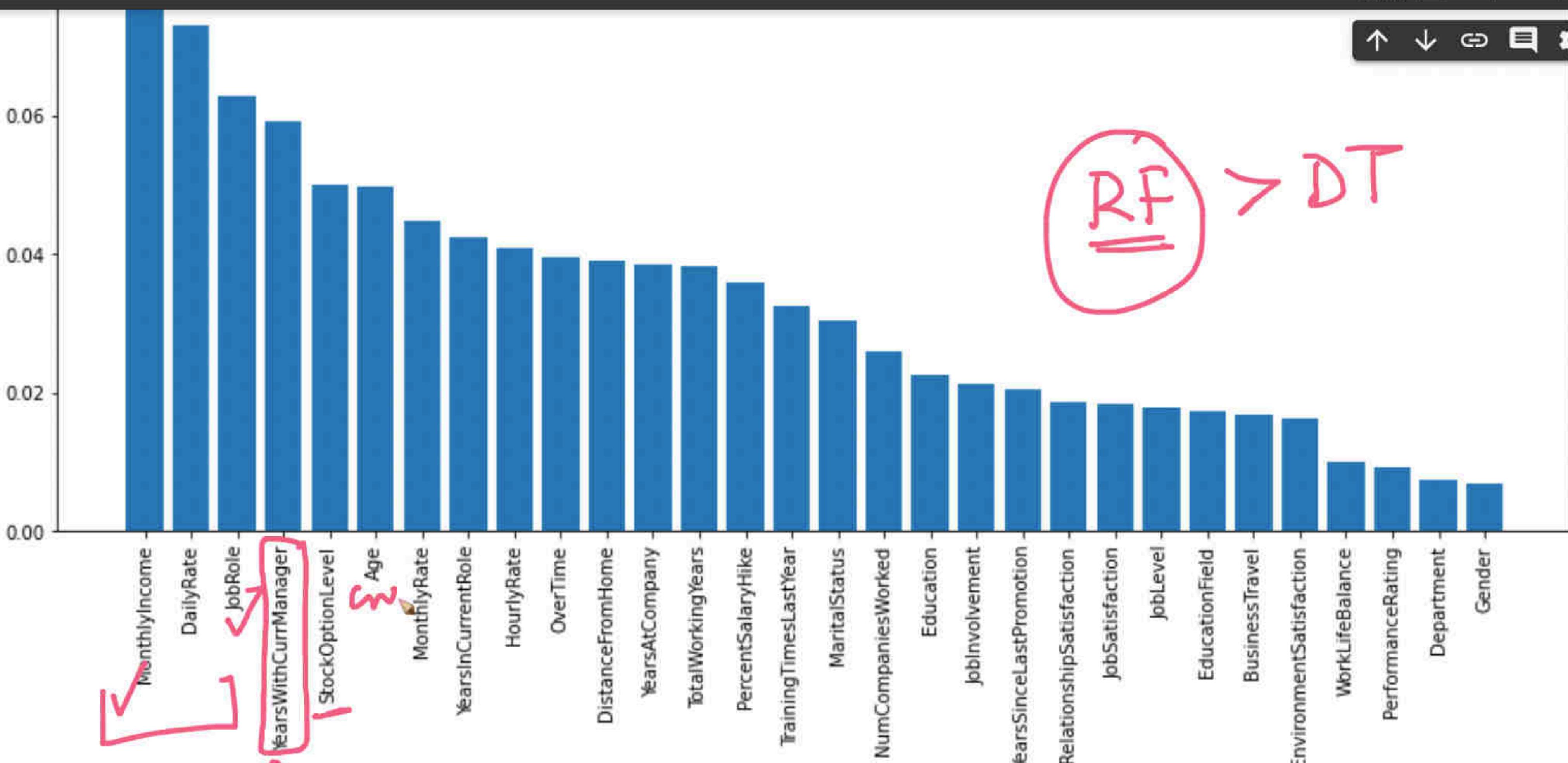
```
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot
```

Feature Importance

94 / 96

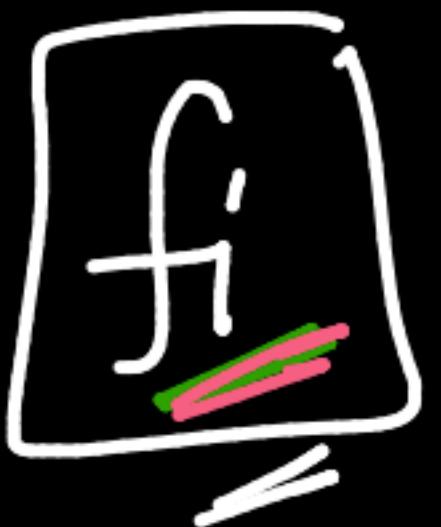


Other Concepts (covered after boosting)

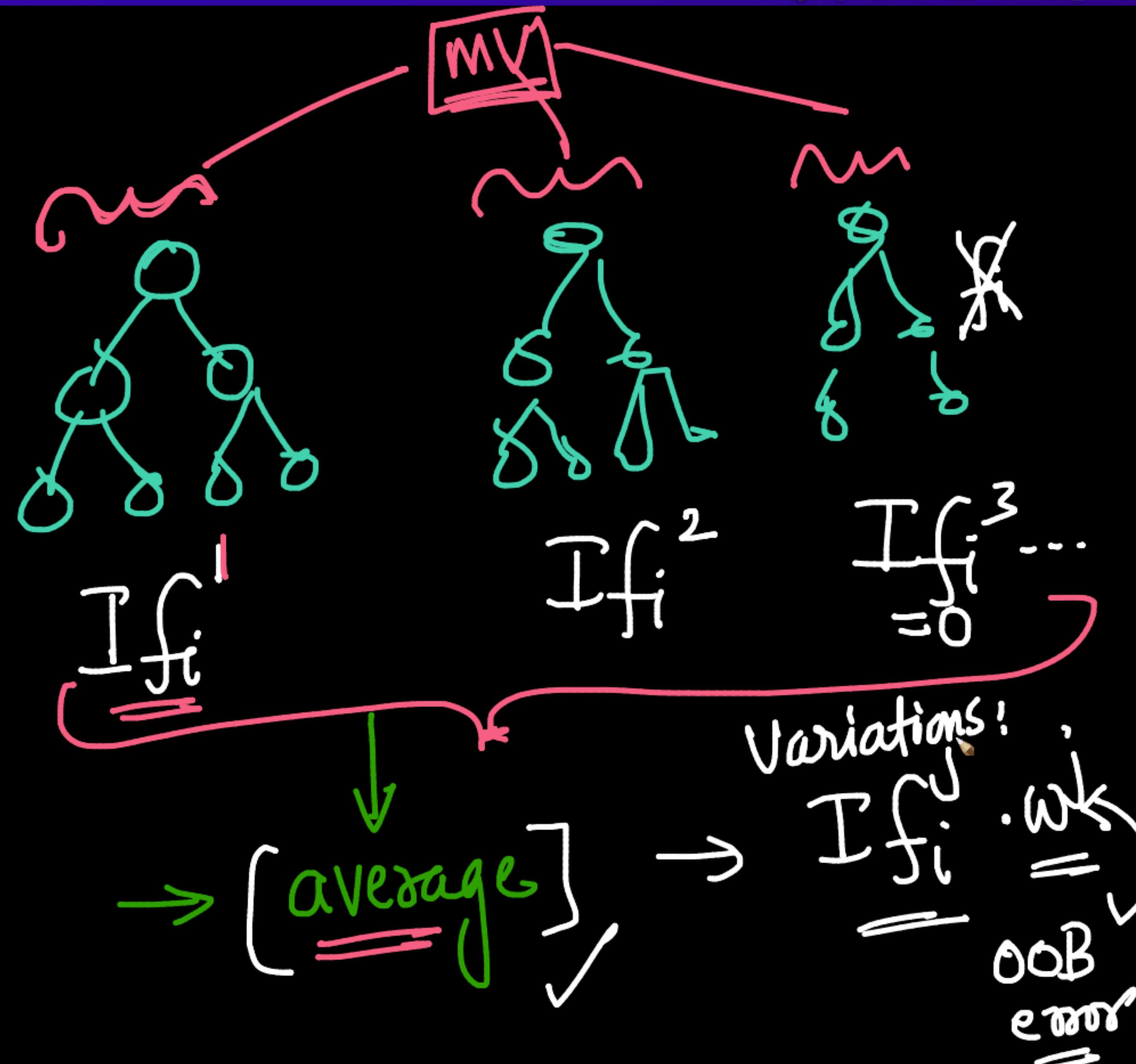


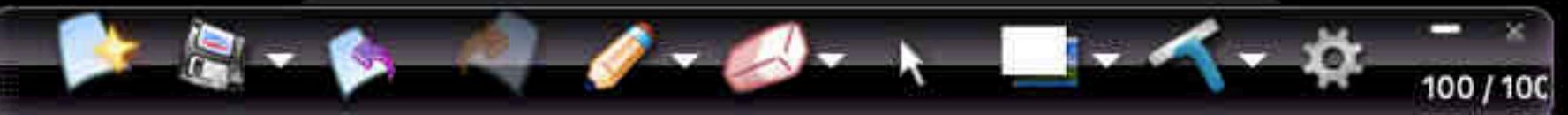
Other Concepts (covered after boosting)

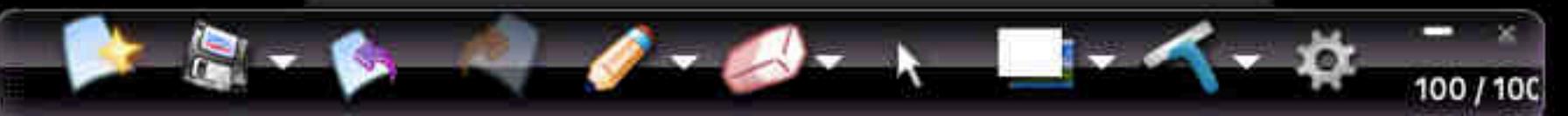
F.I in RF



col-sampling







If $f_{10} \neq 0$

f_{10}

~~100~~ 10

If $f_{12} \neq 0$

f_{12}

~~10~~ 10

f_{10}

$\underline{\underline{CSY}} = 0.1$

~ 10 non-zero

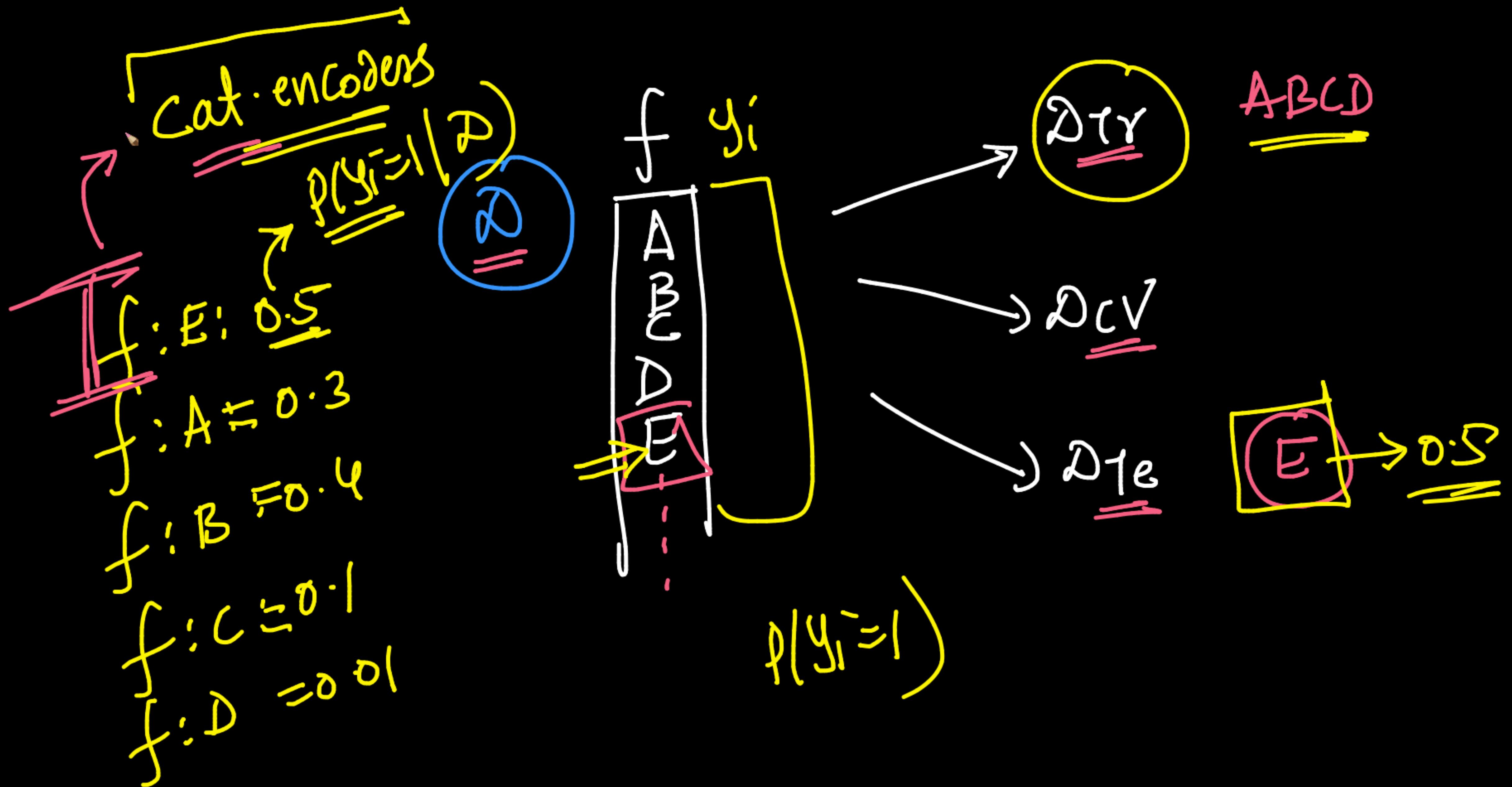
100 Trees

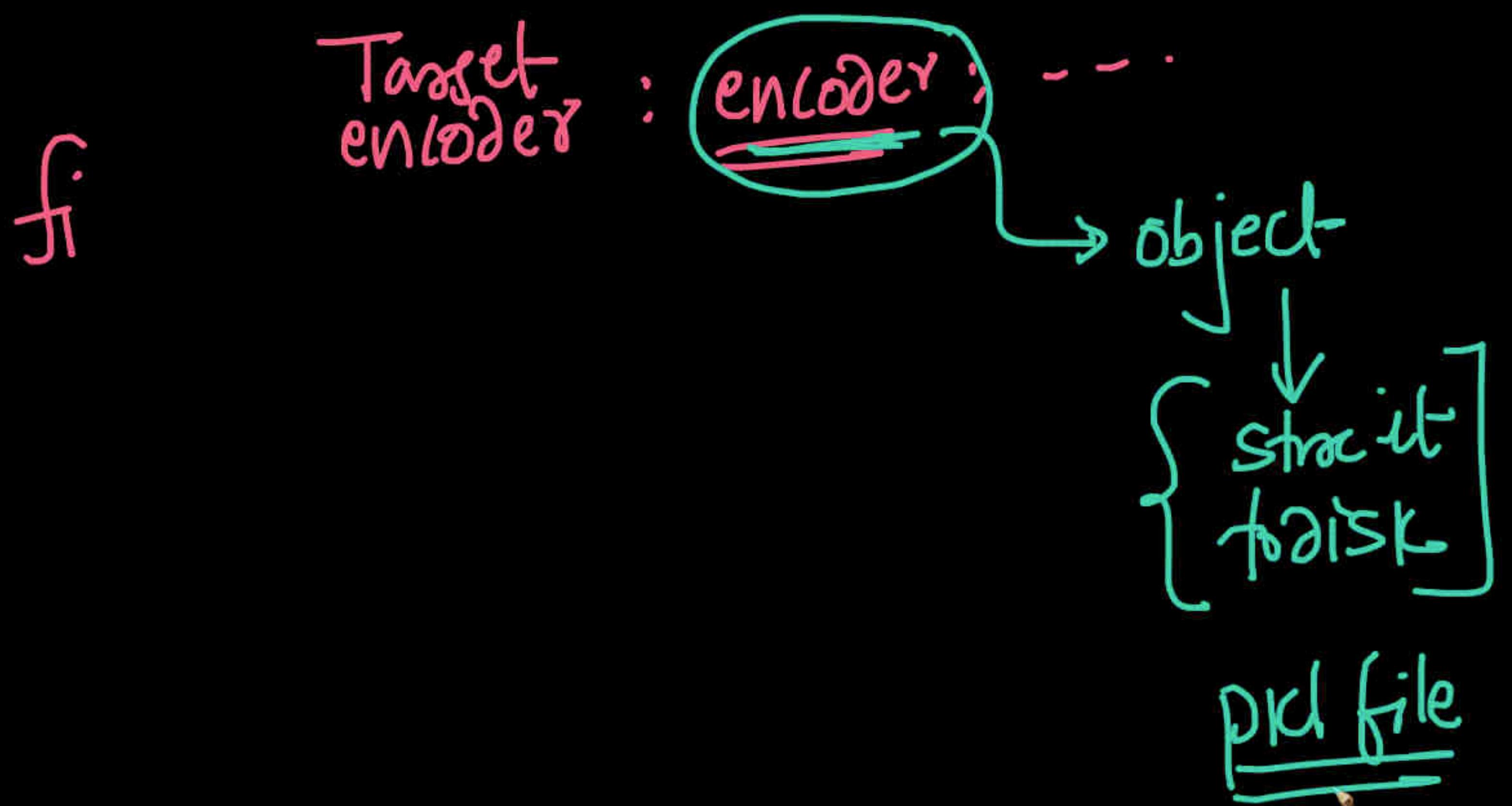
90
Zero

f_{12}

~ 10 non-zero

90
Zero





objects → pkl Python's
optimized
format

JSON
XML ...



Xgboost: Combines both
↓
(lately)

Boosting

↓
overflows to the
next-class ...

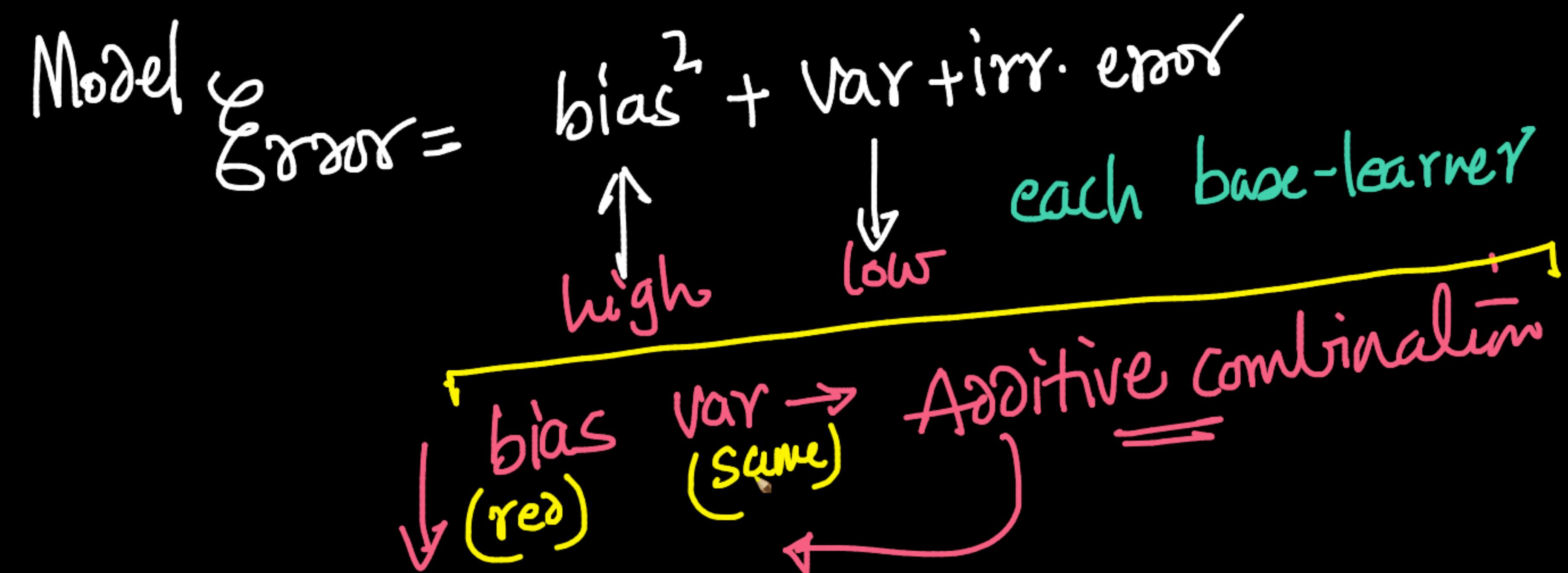
Bagging:

high-var & low bias
base-learners + randomization (S/RS/C) + agg
(deep-trees)

Boosting:
↓

base-learners
(low-var & high-bias)
underfit
[shallow-trees] Decision Stumps

+ additive combination
↓
(few-mins)



Core-Idea:

Frame-work

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

real

easier to grasp
 (regression)

Mean-model

Stage 0: $\hat{M}_0 =$

$$\hat{y}_i = \text{Mean}(y_i)$$

$y_i - \hat{y}_i = \text{error}_i \quad \forall i \in \{1 \dots n\}$

$\hat{h}_0(x)$

Classif: post
read

Stage 1:

M_1

$\{x_i, \text{err}_i\}$

Shallow DT (Decision Stump)
extreme:

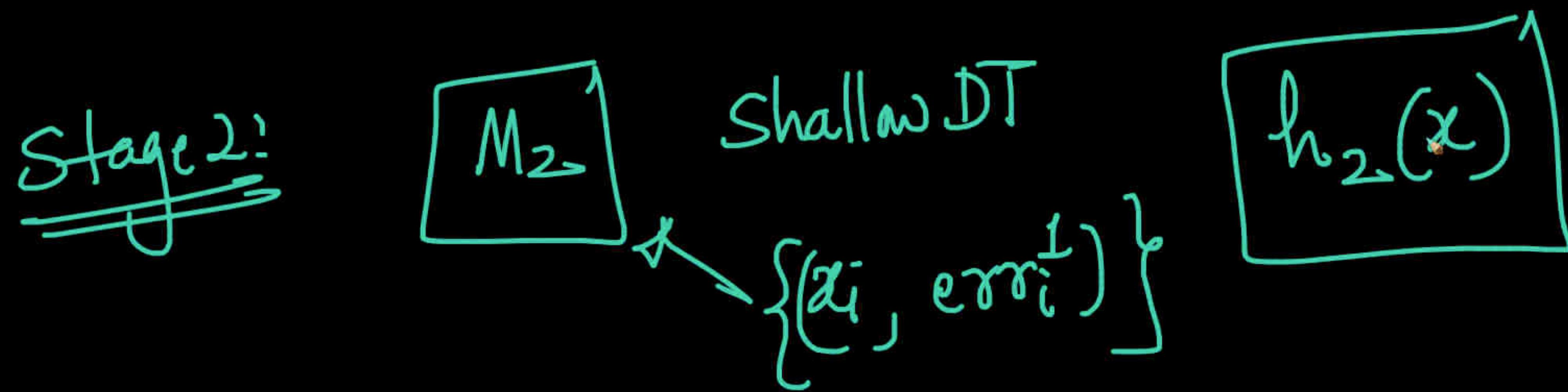
$\underline{h_1(x)}$

Model
after
Stage 1

$$\underline{F_1(x)} = \underline{\alpha_0 h_0(x)} + \underline{\alpha_1 h_1(x)}$$

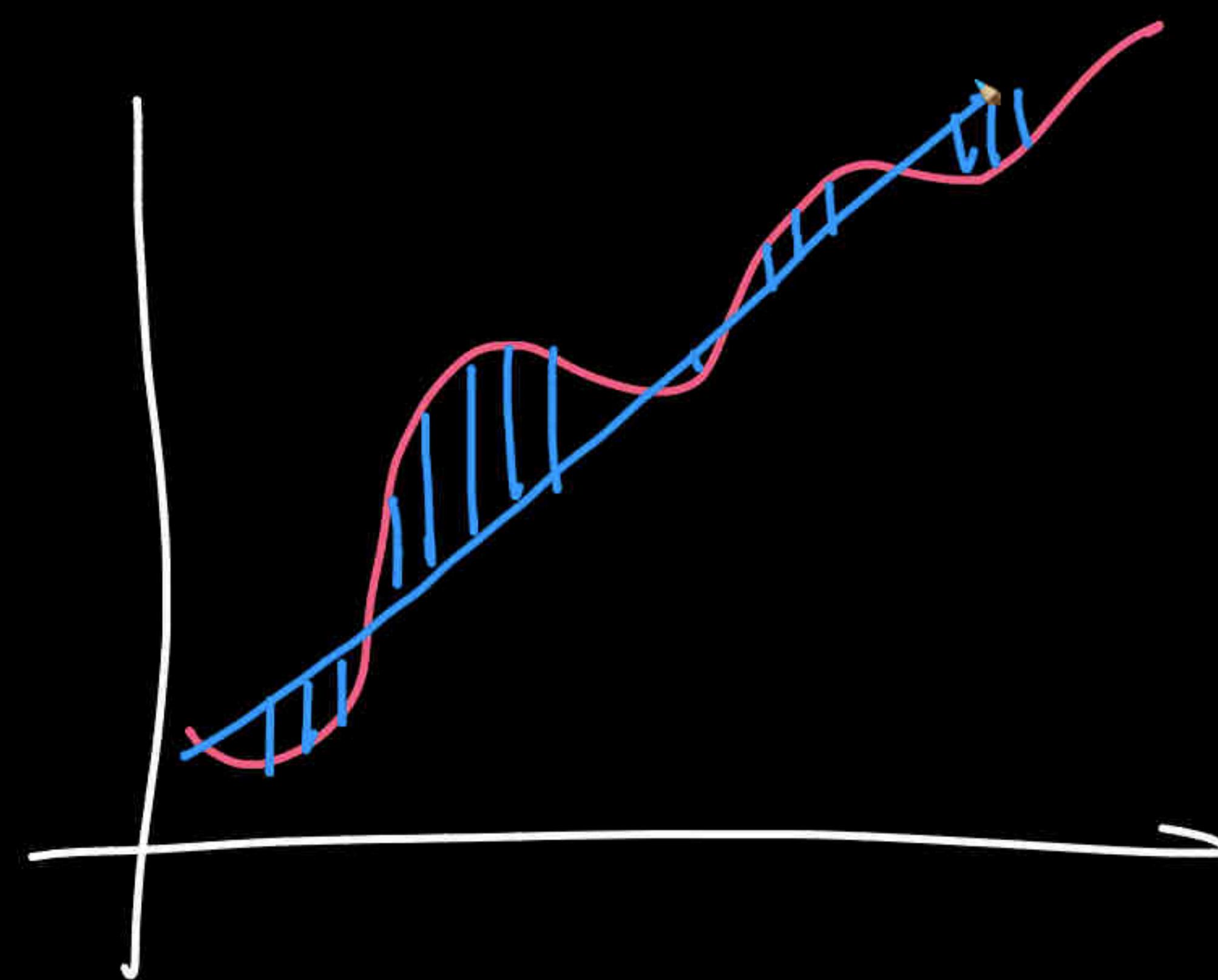
algo to find
 $\underline{\alpha_0}$ & $\underline{\alpha_1}$

$$y_i - f_1(x_i) = \text{err}_i^1 \quad \forall i : 1 \rightarrow n$$



$$F_2(x) = \underline{\alpha_0 h_0(x)} + \underline{\alpha_1 h_1(x)} + \underline{\alpha_2 h_2(x)}$$

Model after
stage 2



repeat ...

{ M-Stages!

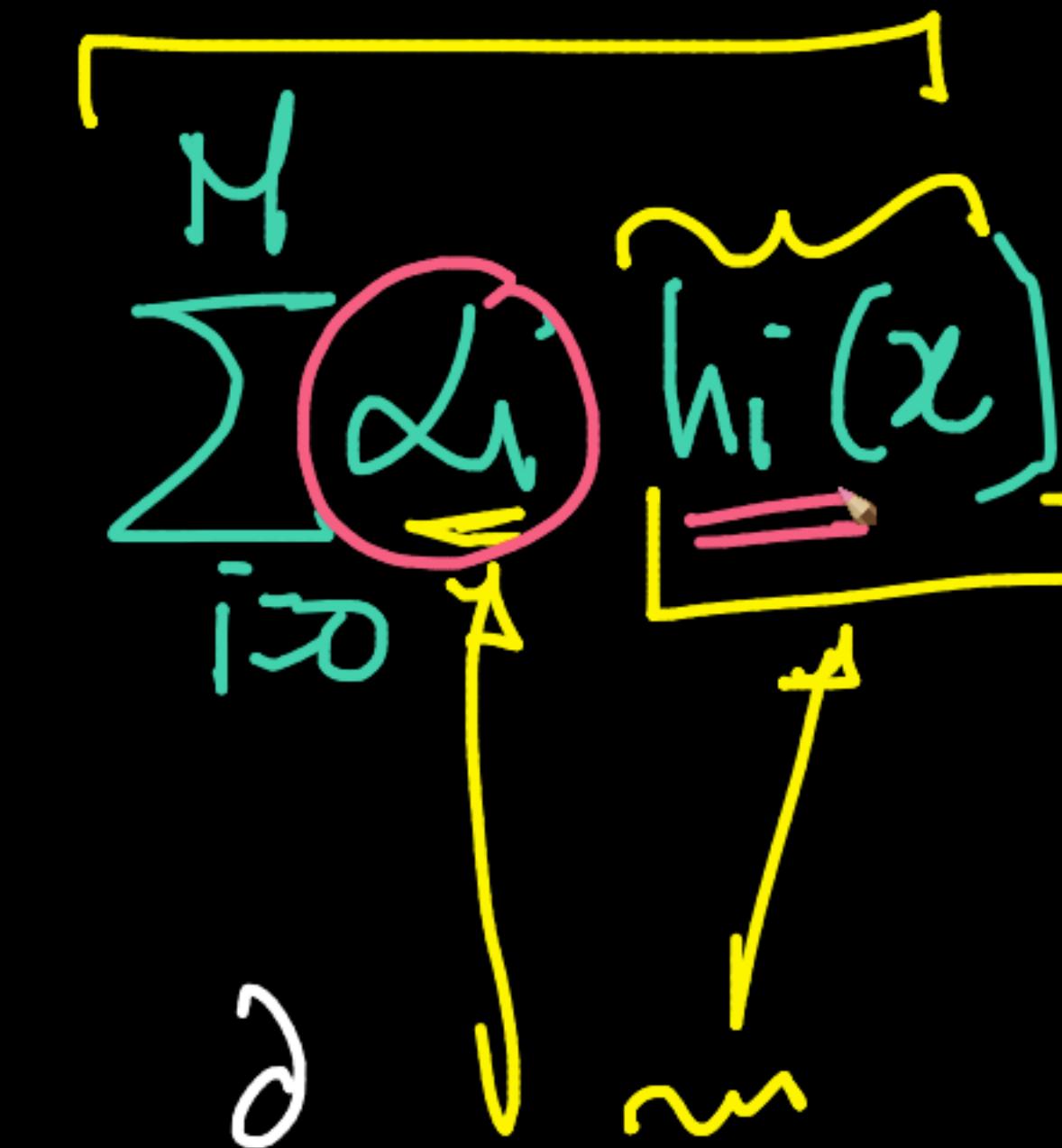
$$F_M(x) = \sum_{i=0}^M d_i h_i(x)$$

mean for $h_0(x)$
shallow DT

optimization to find d_i

Boosting :- learnt
each time @ different
time

$$F_M(x) =$$



Boosting
additive weighted
model

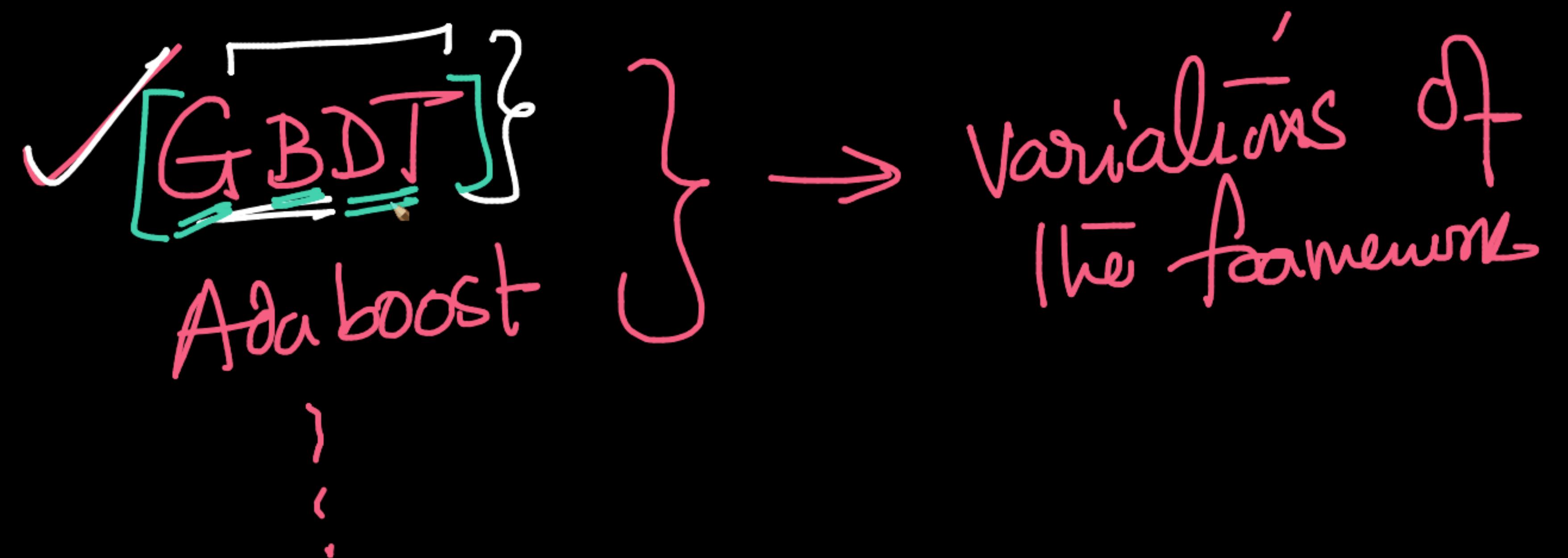
(g. reg):- $F(x) = \sum_{i=1}^d w_i x_i + w_0$

A graph of a linear function $F(x) = \sum w_i x_i + w_0$. The curve starts at a point on the y-axis labeled w_0 and increases with a constant slope, passing through points corresponding to the terms $w_i x_i$.

GD: all w_i 's are learnt @ same time

err_j = errors after stage j

↑
("residuals")

 Variations of
the framework

~~detour~~

$$f_K(x) = \sum_{i=0}^k \alpha_i h_i(x)$$

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

Residuals

Loss-func:

Gradients

Pseudo-residuals

G
B
D
T

[any diff.
loss-function]

[SQ-loss : reg]

[Poisson log-loss : classfn]

w_j → Lr. reg → Sq. loss
→ Logistic reg → log-loss
(later) SVM → hinge loss

e.g.: SQ-loss

$$\underset{\equiv}{\mathcal{L}} \left(\underset{\text{actual}}{y_i}, \underset{\text{predicted}}{\hat{y}_i} \right) = \left(y_i - \hat{f}_k(x_i) \right)^2$$

$$\begin{aligned} \frac{\partial \underset{\text{SQ}}{\mathcal{L}}}{\partial \hat{f}_k(x_i)} &= (-1) \cdot 2 \cdot (y_i - \hat{f}_k(x_i)) \\ &= -2(y_i - \hat{f}_k(x_i)) \end{aligned}$$

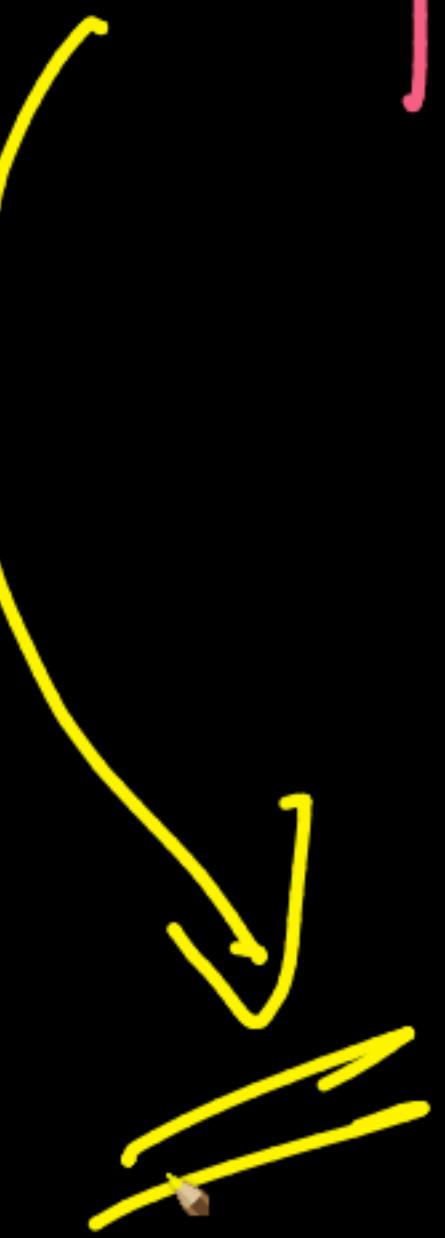
$$\rightarrow \frac{\partial \mathcal{L}^{\text{SO}}}{\partial F_k(x_i)} = 2(y_i - f_k(x_i))$$

y_{ik}

$\text{error}_i^k = \text{res}_{i,k}$

pseudo residual
neg. grad = residual

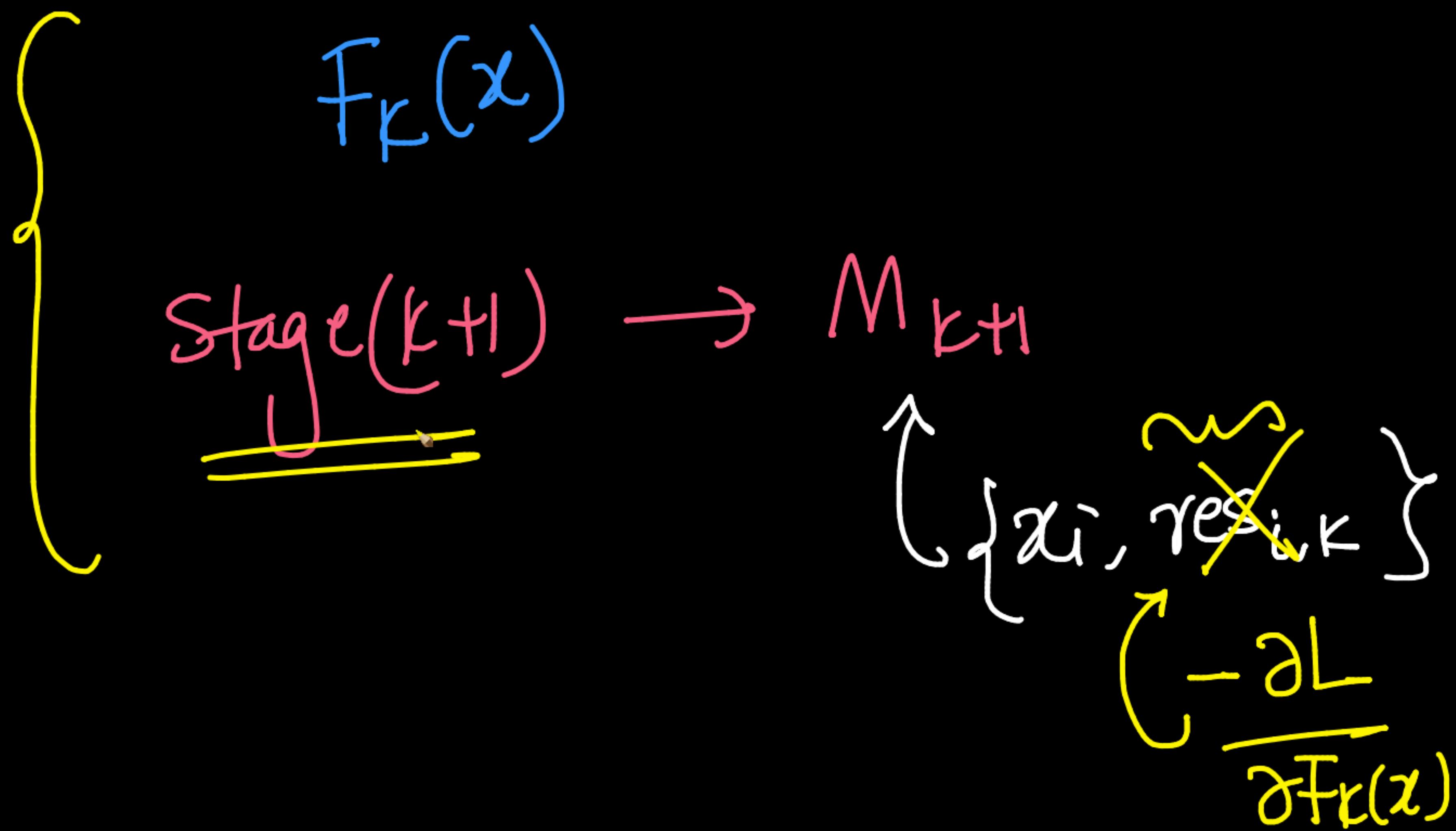
Can prove for log-loss also \rightarrow little more
Math -
(Classifn)



$$\text{neg. grad} = \frac{-\partial \underset{\text{log}}{L}}{\partial f_{\theta}(x_i)} = y_i - p_i$$

\mathcal{L} : differentiable-loss-fn \rightarrow err

$$\frac{-\tilde{\partial} \mathcal{L}}{\partial \hat{f}_r(x_i)} = \underline{\text{residual}_{i,k}}$$



To train $k+1$ base-models:



$$\left\{ \hat{x}_i \rightarrow \frac{\partial L}{\partial F_k(x_i)} \right\}$$

training on residuals

≈ training on
pseudoresiduals

Boosting



neg. grad as pseudo residual



instead of the
residual/error

↓
{\$\underline{G}\underline{BDT}\$}

→ why is this popular

↳ Loss fn of my choice
Flexibility

(later)

SUM → hinge-loss

Logistic reg → ~~log-loss~~

(later)

Softmax → CE loss

~~log-loss~~ $\rightarrow \{0, 0.9, 0.1\}$

penalize
incorrect

1 : 2
0.1 : 3
- - -
- - -

Reg

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

Sq-loss \rightarrow lr. reg

{

Abs. loss

$$(y_i - \hat{y}_i)^4$$



\rightarrow Much more
penalization of
errors

however, i.e. when the set \mathcal{H} is finite, we choose the candidate function h closest to the gradient of L for which the coefficient γ may then be calculated with the aid of [line search](#) on the above equations. Note that this approach is a heuristic and therefore doesn't yield an exact solution to the given problem, but rather an approximation. In pseudocode, the generic gradient boosting method is:^{[5][2]}

Input: training set $\{(x_i, y_i)\}_{i=1}^n$, a differentiable loss function $L(y, F(x))$, number of iterations M .

Algorithm:

1. Initialize model with a constant value:

$$F_0(x) = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, \gamma).$$

2. For $m = 1$ to M :

1. Compute so-called *pseudo-residuals*:

$$r_{im} = - \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \dots, n.$$

2. Fit a base learner (or weak learner, e.g. tree) closed under scaling $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^n$.

3. Compute multiplier γ_m by solving the following [one-dimensional optimization](#) problem:

$$\gamma_m = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)).$$

4. Update the model:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$

3. Output $F_M(x)$.

en.wikipedia.org/wiki/Gradient_boosting

solution to the given problem, but rather an approximation. In pseudocode, the generic gradient boosting method is:^{[5][2]}

Input: training set $\{(x_i, y_i)\}_{i=1}^n$, a differentiable loss function $L(y, F(x))$, number of iterations M .

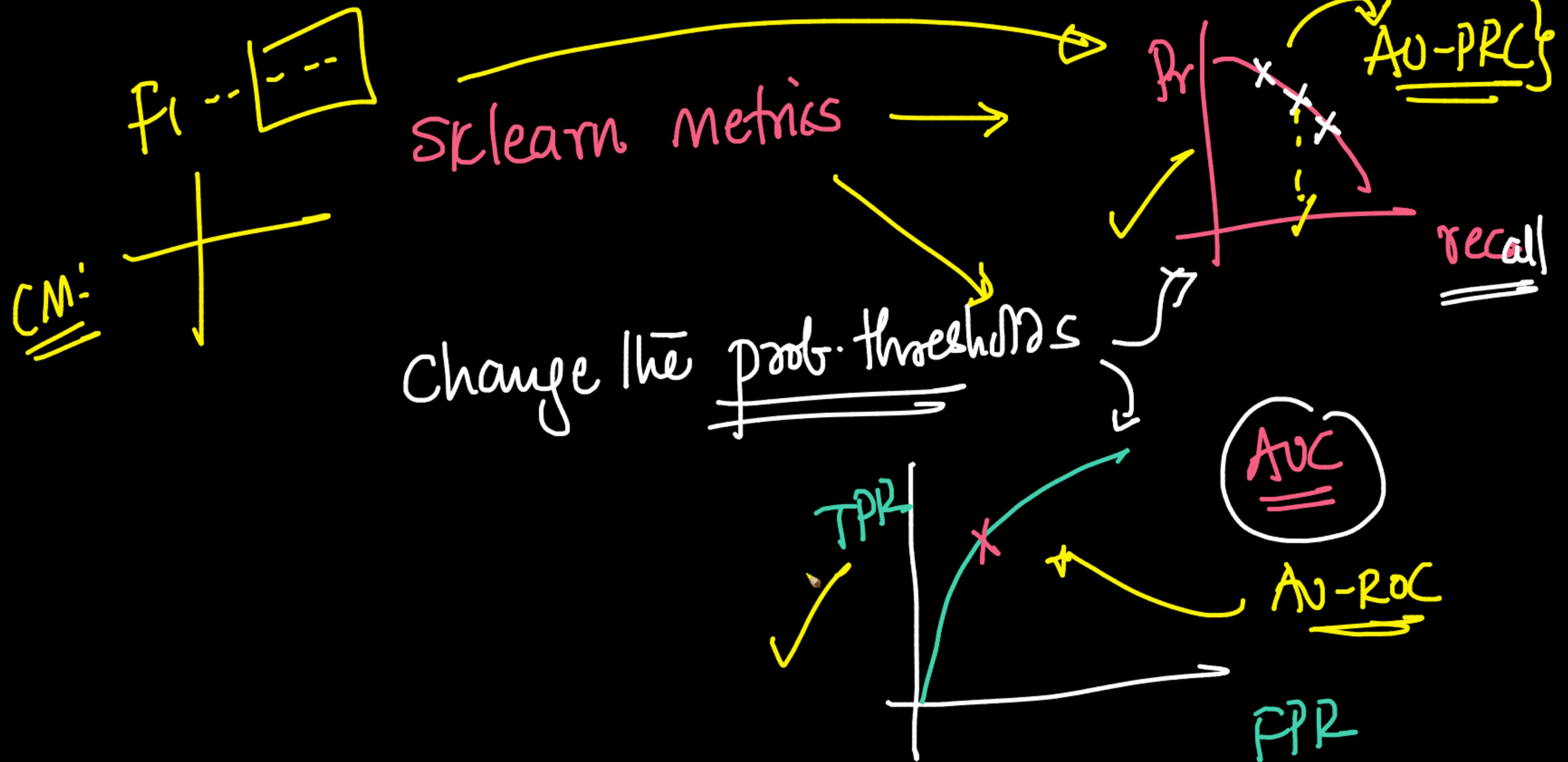
Algorithm:

1. Initialize model with a constant value:
$$F_0(x) = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, \gamma).$$
2. For $m = 1$ to M :
 1. Compute so-called *pseudo-residuals*:
$$r_{im} = - \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \dots, n.$$
 2. Fit a base learner (or weak learner, e.g. tree) closed under scaling $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^n$.
 3. Compute multiplier γ_m by solving the following [one-dimensional optimization](#) problem:
$$\gamma_m = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)).$$
 4. Update the model:
$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$
3. Output $F_M(x)$.

Amazon

Gradient tree boosting [edit]

Gradient boosting is typically used size as base learners. For this special case,



$$(x_i, y_i) \xrightarrow{\text{mean}} h_0(x) \xrightarrow{\text{err}_i^0} h_1(x); \quad \alpha_0 h_0(x) + \alpha_1 h_1(x) = F_1(x)$$

$$\text{err}_i^j = y_i - f_j(x)$$

$$\downarrow \text{err}_i^1 = h_2(x)$$

$$\alpha_0 h_0(x) + \alpha_1 h_1(x) + \alpha_2 h_2(x)$$

⋮
M basis-funs

DT and RF.ipynb - Colaboratory | sklearn.tree.DecisionTreeClass... | sklearn.ensemble.RandomFore... | sklearn.preprocessing.LabelEnc... | Target Encoder — Category Enc... | Imbalanced Data.ipynb - Colab... | Gradient boosting - Wikipedia | +

colab.research.google.com/drive/1d2HJ9AKulf83i7pwmDrL-2FOCncFubRF#scrollTo=Pxe9cuS7qp8U

+ Code + Text

RAM Disk ✓

[239] clf = DecisionTreeClassifier(random_state=0, max_depth=l_best, class_weight={ 0:0.1, 1:w })

[239] clf.fit(X_train, y_train)

y_pred_val = clf.predict(X_val)

val_score = f1_score(y_val, y_pred_val)

print(val_score)

confusion_matrix(y_val, y_pred_val)

2
0.41618497109826585
array([[157, 87],
 [14, 36]])

[101] # Feature importance

```
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importance
plt.figure(figsize=(15, 7)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot
```



131 / 131