

<https://stackoverflow.com/questions/50176096/removing-redundant-columns-when-using-get-dummies> (<https://stackoverflow.com/questions/50176096/removing-redundant-columns-when-using-get-dummies>)

```
df
Out[16]:
```

	gender	eyes
0	male	blue
1	female	brown
2	male	black

using the function get_dummies I get the following dataframe

```
df_dummies = pandas.get_dummies(df)
```

```
df_dummies
Out[18]:
```

	gender_female	gender_male	eyes_black	eyes_blue	eyes_brown
0	0	1	0	1	0
1	1	0	0	0	1
2	0	1	1	0	0



However the columns gender_female and gender_male contain the same information because

In []:

```
df
Out[16]:
```

	gender	eyes
0	male	blue
1	female	brown
2	male	black

using the function get_dummies I get the following dataframe

```
df_dummies = pandas.get_dummies(df)
```

```
df_dummies
Out[18]:
```

	gender_female	gender_male	eyes_black	eyes_blue	eyes_brown
0	0	1	0	1	0
1	1	0	0	0	1
2	0	1	1	0	0

Out of n-cols, only (n-1) can be used. Adv: Reduces Multicollinearity. ($X \sim X$)



However the columns gender_female and gender_male contain the same information because

In []:

<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html> (<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html>)

[learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html\)](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html)

Examples

LabelEncoder can be used to normalize labels.

```
>>> from sklearn import preprocessing
>>> le = preprocessing.LabelEncoder()
>>> le.fit([1, 2, 2, 6])
LabelEncoder()
>>> le.classes_
array([1, 2, 6])
>>> le.transform([1, 1, 2, 6])
array([0, 0, 1, 2]...)
>>> le.inverse_transform([0, 0, 1, 2])
array([1, 1, 2, 6])
```

It can also be used to transform non-numerical labels (as long as they are hashable and comparable) to numerical labels.

```
>>> le = preprocessing.LabelEncoder()
>>> le.fit(["paris", "paris", "tokyo", "amsterdam"])
LabelEncoder()
>>> list(le.classes_)
['amsterdam', 'paris', 'tokyo']
>>> le.transform(["tokyo", "tokyo", "paris"])
array([2, 2, 1]...)
>>> list(le.inverse_transform([2, 2, 1]))
```

Handwritten annotations:

- A yellow box encloses the first table. Inside, 'Animal' is written above the first column, and 'cat', 'dog', 'cat', 'dog', 'elephant', 'lion', 'elephant' are listed in the second column.
- A pink arrow labeled 'Encoding' points from the first table to the second table.
- The second table has 'Animal' at the top and contains the numbers 0, 1, 0, 1, 2, 3, 2 in its second column.
- A blue button in the top right corner says 'Hide prompts and outputs'.
- Below the code cell, there is a small video thumbnail of a person's face.

```
In [1]: from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

le.fit(["cat", "dog", "cat", "dog", "elephant", "lion", "elephant"])

list(le.classes_)
```

Out[1]: ['cat', 'dog', 'elephant', 'lion']

```
In [2]: le.transform(["cat", "dog", "cat", "dog", "elephant", "lion", "elephant"])
```

Out[2]: array([0, 1, 0, 1, 2, 3, 2])

```
In [3]: list(le.inverse_transform([0, 1, 0, 1, 2, 3, 2]))
```

Out[3]: ['cat', 'dog', 'cat', 'dog', 'elephant', 'lion', 'elephant']

Age Salary

```
le.fit(["cat", "dog", "cat", "dog", "elephant", "lion", "elephant"])
list(le.classes_)

Out[1]: ['cat', 'dog', 'elephant', 'lion']

In [2]: le.transform(["cat", "dog", "cat", "dog", "elephant", "lion"])
Out[2]: array([0, 1, 0, 1, 2, 3, 2]) 20 Unique
35 | Lakh ①---②---③---④---⑤---⑥---⑦---⑧---⑨
34 | 80000: list(le.inverse_transform([0, 1, 0, 1, 2, 3, 2]))

Out[3]: ['cat', 'dog', 'cat', 'dog', 'elephant', 'lion', 'elephant']

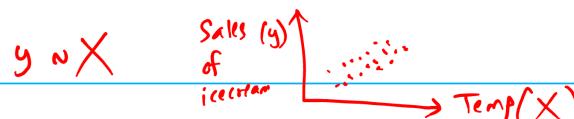
In [ ]: ShirtSize (Ordinal)
S — 0
M — 1
L — 2
XL — 3
S — 0
```

+5 +5 1(8) VAR



In []:

5. R – Squared



- It explains proportion of the variance in the dependent variable that is predictable from the independent variable(s).
- Also known as Coefficient of Determination and value lies between 0 & 1 (higher, the better).
- Shortcoming: It always increases with addition of more features.

$$[0-1] \quad 1 - \left[\frac{RSS}{TSS} \right]$$

$$R^2 = 1 - \frac{RSS}{TSS}$$

$$\sum_{i=1}^n (y_i - \bar{y})^2 = (y_1 - \bar{y})^2 + (y_2 - \bar{y})^2 + \dots + (y_n - \bar{y})^2$$

$$= e_1^2 + e_2^2 + \dots + e_n^2$$

Residual Sum of Squares(RSS) = $e_1^2 + e_2^2 + \dots + e_n^2$

Total Sum of Squares(TSS) = $\sum (y_i - \bar{y})^2$

- y_i = Actual value
- \bar{y} = Mean value
- e = Error

mean of
actual y .



©INSAID All rights reserved.



In []:

R – Squared Example

X_1	X_2	X_3	y_i	\hat{y}_i
TV	Radio	Newspaper	Sales(actual)	Sales(predicted)
230	40	70	22	23.5
45	42	45	10	12
20	45	60	15	14.2
150	41	55	20	17.3
180	12	30	25	24.9

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

$$\text{RSS} = (22 - 23.5)^2 + (10 - 12)^2 + (15 - 14.2)^2 + (20 - 17.3)^2 + (25 - 24.9)^2 = 14.19$$

$$\text{TSS} = (22 - 18.4)^2 + (10 - 18.4)^2 + (15 - 18.4)^2 + (20 - 18.4)^2 + (25 - 18.4)^2 = 141.2$$

$$R^2 = 1 - \frac{14.19}{141.2}$$

$$R^2 = 1 - 0.100$$

$$\text{Mean} = 18.4$$

$$\therefore R^2 = 0.9$$

©INSAID All rights reserved.



In []:

11

12

13

①

$A' \quad P_1 \quad P_2 \quad P_3 \quad P_4$

$+4 \quad +1 \quad -4 \quad -4$

$P_1 \quad P_2 \quad -4 \quad -4$

n

$ME = \frac{\sum_{i=1}^n (A_i - P_i)}{n}$

$A \quad P_1 \quad P_2 \quad P_3 \quad P_4$

$0 \quad 0 \quad 0 \quad 0$

$ME = \frac{0+0+0+0}{4} = 0$

Mean error: $\frac{(4) + (+) + (-4) + (-4)}{4} = 0$

Mean Absolute error: $\frac{\sum_{i=1}^n |A_i - P_i|}{n}$

②

$+7 \quad +1 \quad -1 \quad -2$

$|+7| + |+1| + |-1| + |-2| = 16$

$MAE = \frac{|+7| + |+1| + |-1| + |-2|}{4} = 4$



In []:

You are screen sharing

① $MAE = \frac{|+4| + |+4| + |-4| + |-4|}{4} = \frac{16}{4} = 4$

② $MAE = \frac{|+7| + |+1| + (-6) + (-2)}{4} = \frac{7 + 1 - 6 - 2}{4} = \frac{-2}{4} = -0.5$

Spread of errors $\textcircled{2} > \textcircled{1}$

Mean Squared Error = $\frac{\sum_{i=1}^n (A_i - P_i)^2}{n}$

In []:

You are screen sharing

Mean Squared Error = $\frac{\sum_{i=1}^n (A_i - P_i)^2}{n}$

① $MSE = \frac{(+4)^2 + (+4)^2 + (-4)^2 + (-4)^2}{4} = \frac{16}{4} = 4$

② $MSE = \frac{(+7)^2 + (+1)^2 + (-6)^2 + (-2)^2}{4} = \frac{90}{4} = 22.5$

Root MSE = \sqrt{MSE}

① $\sqrt{16} = 4$

② $\sqrt{22.5} = 4.74$

X y
Train 160
Test 20 x_{train} y_{train} x_{test} y_{test}
 fit

In []:

The screenshot shows a Jupyter Notebook interface with a dark theme. On the left, there are four small thumbnail images labeled 14, 15, 16, and 17, which appear to be screenshots of other parts of the notebook or related documents. The main content area contains handwritten notes in green ink:

Classification Algorithms:

1. Logistic Regression
2. Native Bages Classifier
3. Support Vector Classifier
4. Decision Tree Classifier
5. Random Forest Classifier
6. k-NN Classifier
7. AdaBoost Classifier
8. Gradient Boost Classifier
9. LightGradientBoost Classifier
10. XGBoost Classifier
11. ExtraTrees Classifier
12. CatBoost Classifier.

Below the notes is a small diagram consisting of three circles with the number '0' next to each one. In the bottom right corner of the main area, there is a video feed of a person wearing glasses and a light-colored shirt.

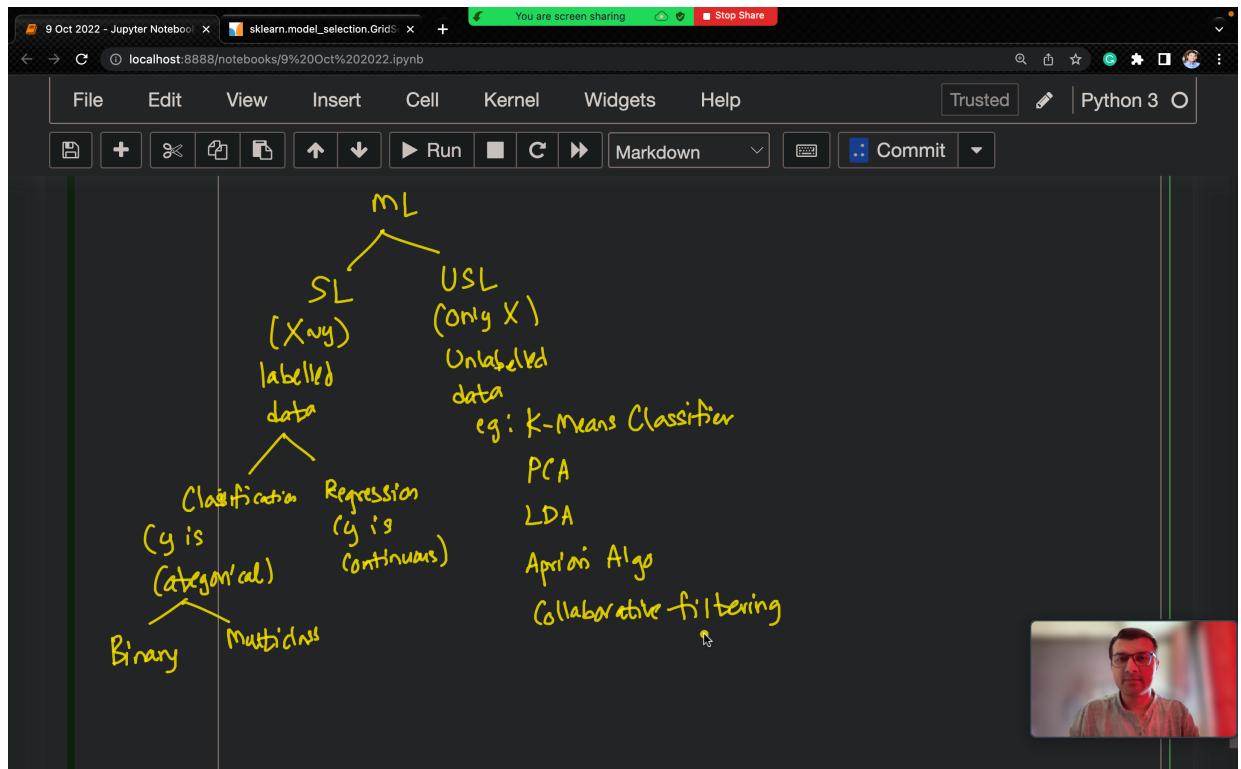
The screenshot shows a Jupyter Notebook interface with a dark theme. The top bar indicates the notebook is titled "9 Oct 2022 - Jupyter Notebook" and the file name is "sklearn.model_selection.GridS.ipynb". The main content area contains handwritten notes in green ink:

Regression:

- Linear Regression
- Decision Tree Regressor
- Random Forest Regressor
- SupportVector Regressor
- GradientBoost Regressor
- ExtraTrees Regressor
- XGBoost Regressor
- Polynomial Regression

In the bottom right corner of the main area, there is a video feed of a person wearing glasses and a light-colored shirt.

In []:



In []:

Happy Learning