

# Intermediate Python & a Glimpse into AI applications

Class 4

Pepe Bonet Giner

16<sup>th</sup> January 2023

# Index Class 4

---

Topic 1: Recap

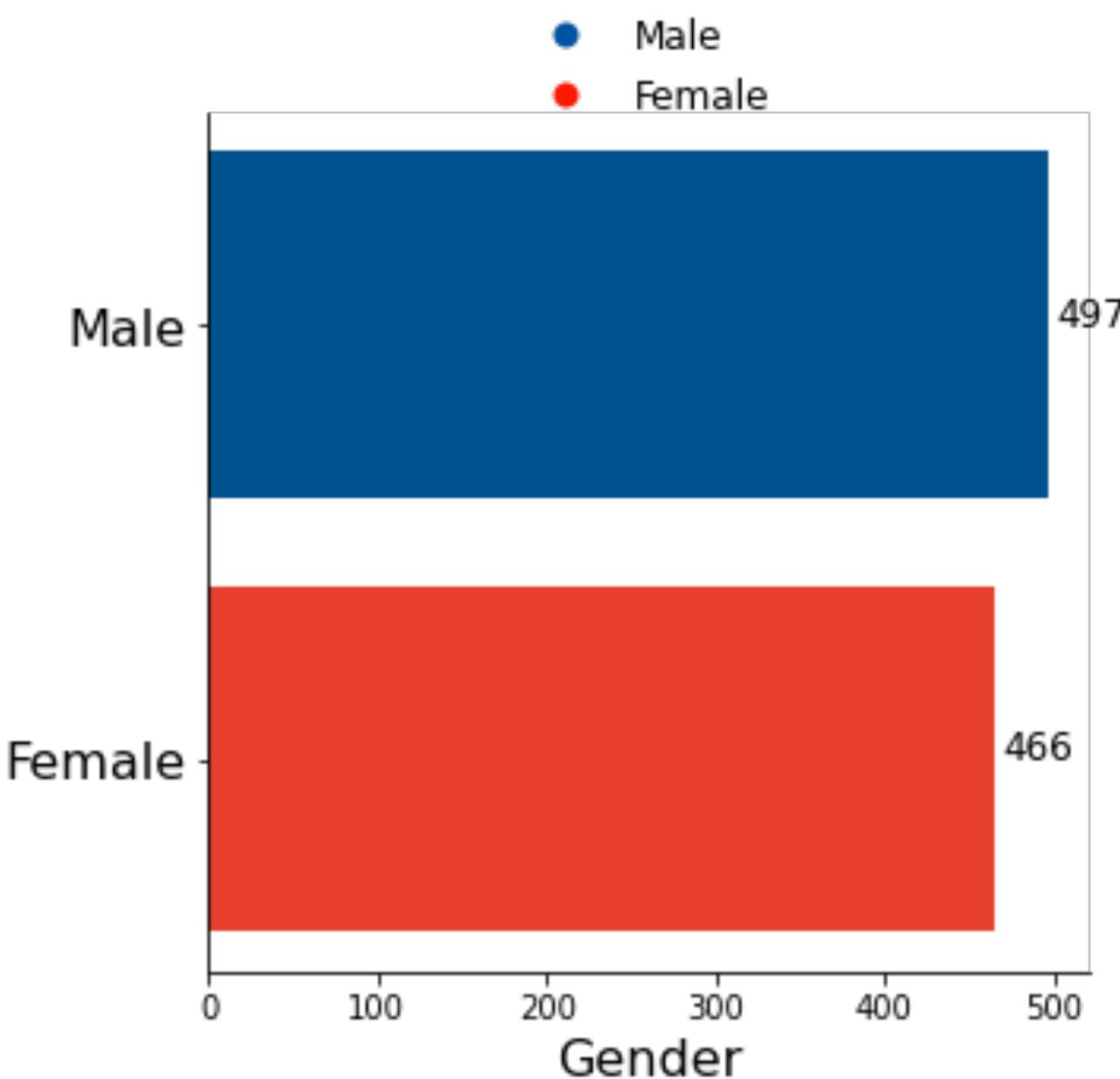
Topic 2: Intro to the world of predictive models

Topic 3: Supervised vs. Unsupervised. Regression vs. Classification

Topic 4: Intro to ML models in Python

# Topic 1: Recap

# Recap: Matplotlib/Seaborn and EDA steps



matplotlib



seaborn

```
#Start Figure
fig, ax = plt.subplots(figsize=(5, 5))

#Body of the figure to build and the data to use
sns.barplot(x=to_plot['gender'], y=to_plot['index'],
             palette=['#08519c', '#f03b20'])

#Change Axes
ax.set_xlabel("Gender", fontsize=16)
ax.set_ylabel("", fontsize=16)
ax.set_yticklabels(['Male', 'Female'], fontsize=16)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

# Add Numbers to plot
for index, row in to_plot.iterrows():
    ax.text(row.gender + 25, index, row.gender,
            color='black', ha="center", fontsize=12)

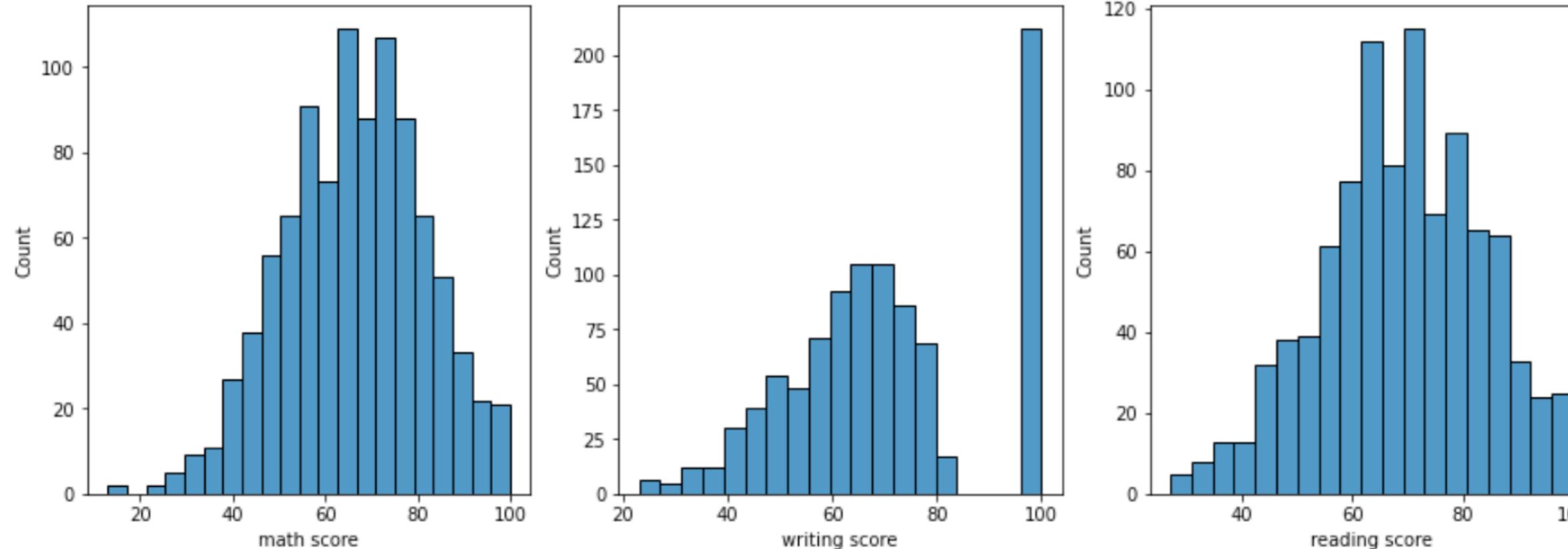
## Add legend
custom_lines = []
for el in [('Male', '#08519c'), ('Female', '#f03b20')]:
    custom_lines.append(
        plt.plot([],[], marker="o", ms=8, ls="", mec='black',
                  mew=0, color=el[1], label=el[0])[0]
    )
ax.legend(
    bbox_to_anchor=(0., 1.05, 1., .102),
    handles=custom_lines, loc='upper center',
    facecolor='white', ncol=1, fontsize=12, frameon=False
)
#Save or show
plt.show()
```

# Recap: EDA & Feature Engineering

---

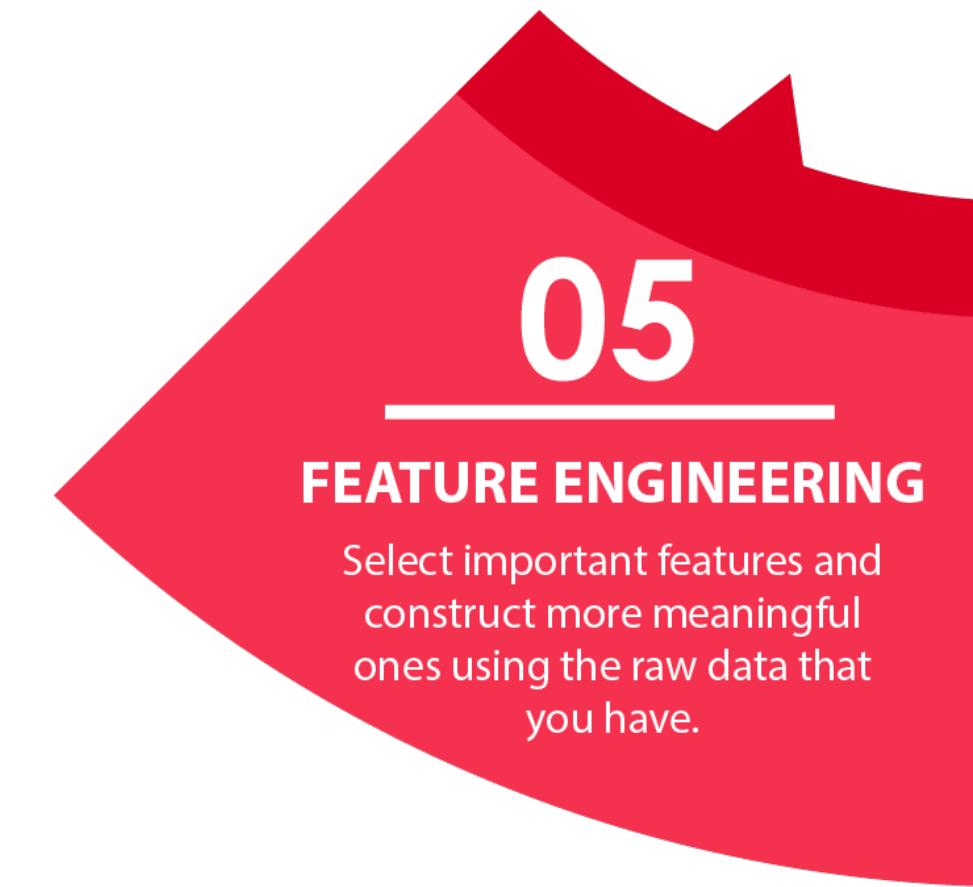
## EDA

- Is a process where users look at and understand their data with statistical and visualization methods
- Categorical EDA
- Numerical EDA



## Feature Engineering

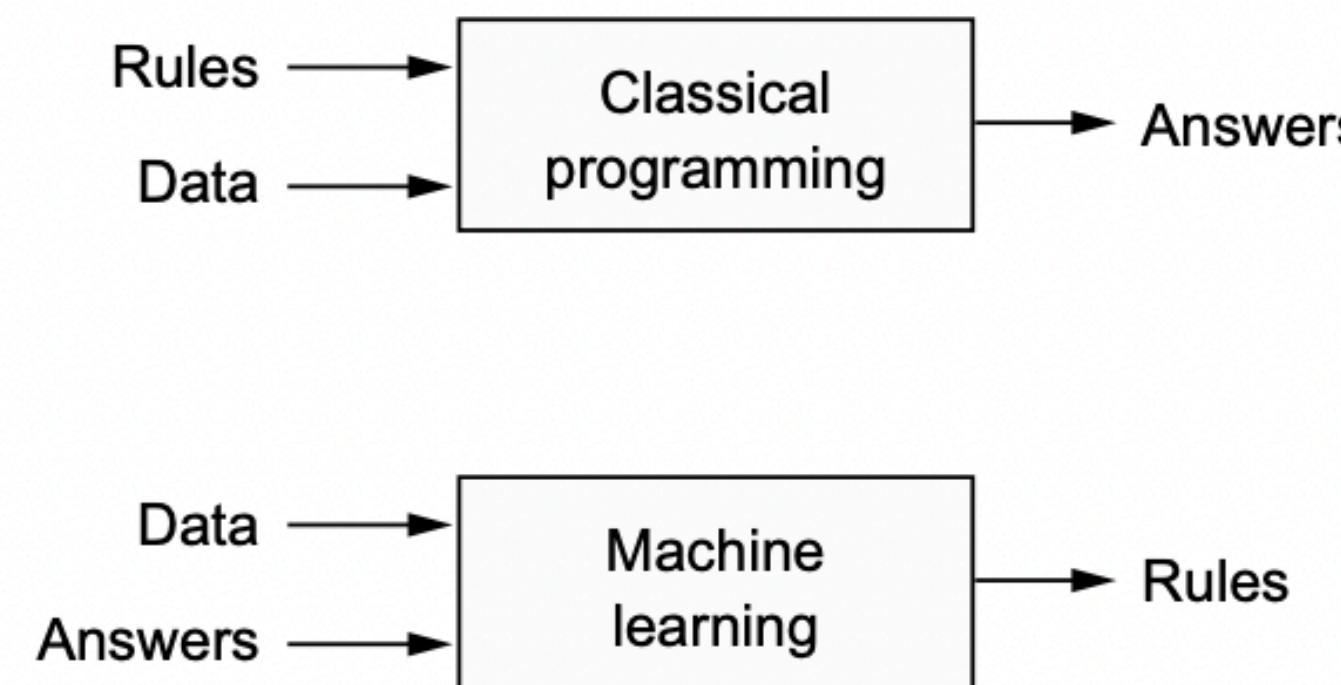
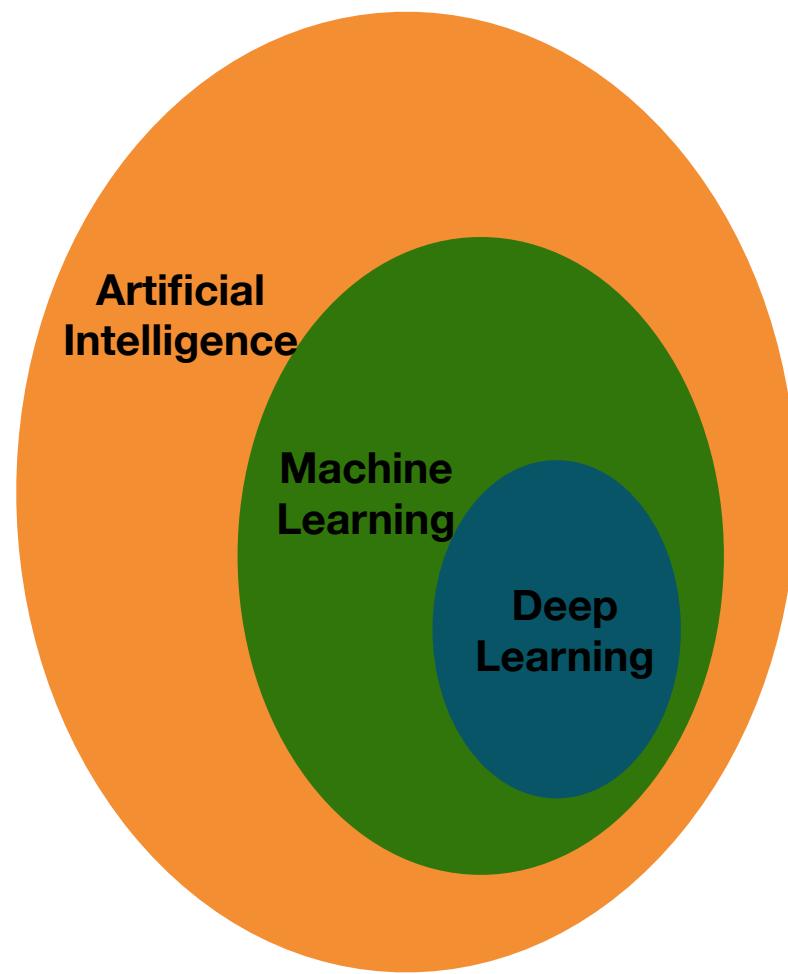
- Is the process of using domain knowledge to extract features (characteristics, properties, attributes) from raw data



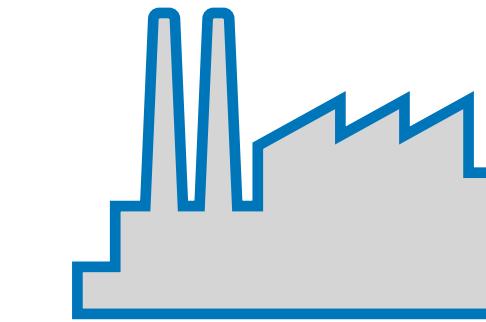
# Topic 2: Intro to the world of predictive models

# A shift of paradigm

---



T<sup>a</sup>, Pressure, Week Day



Rules

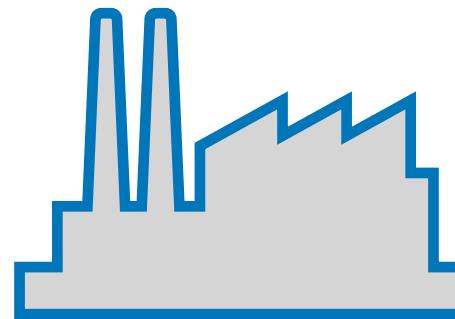
AI/ML

- AI: Effort to automate intellectual tasks normally performed by humans
- ~ 1950-1980 Symbolic AI. Rules + Data → Answers
- ML: Data + Answers → Rules

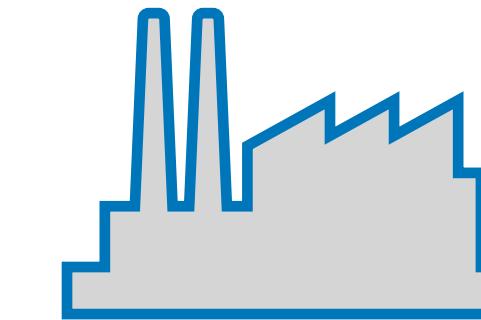
# Learning from the past to predict the future

---

Rules



AI/ML



Collect data throughout time

```
if  $T^a > 50^\circ$ :  
    return "We are in danger!"  
elif  $T^a < 10^\circ$ :  
    return "The tank is too cold!"  
else:  
    return "Everything is fine!"
```

Identifier	Features			Label
ID	$T^a$	Pressure	Week Day	Fault
1	35	10	1	No
2	68	25	2	Yes
3	42	10	1	No
4	15	12	6	No

# Learning from the past to predict the future

---

Identifier	Features			Label
ID	T <sup>a</sup>	Pressure	Week Day	Fault
1	35	10	1	No
2	68	25	2	Yes
3	42	10	1	No
4	15	12	6	No

**Predict Failure**

# Learning from the past to predict the future

---

Identifier	Features			Label
ID	T <sup>a</sup>	Pressure	Week Day	Fault
1	35	10	1	No
2	68	25	2	Yes
3	42	10	1	No
4	15	12	6	No

Predict Failure

ID	#Bedrooms	#bathrooms	M2	Price(Euro)
1	3	2	120	190.000
2	3	3	150	250.000
3	1	1	60	100.000
4	2	2	80	140.000

Predict House Prices

# Learning from the past to predict the future

---

Identifier

Features

Label

ID	T <sup>a</sup>	Pressure	Week Day	Fault
1	35	10	1	No
2	68	25	2	Yes
3	42	10	1	No
4	15	12	6	No

Predict Failure

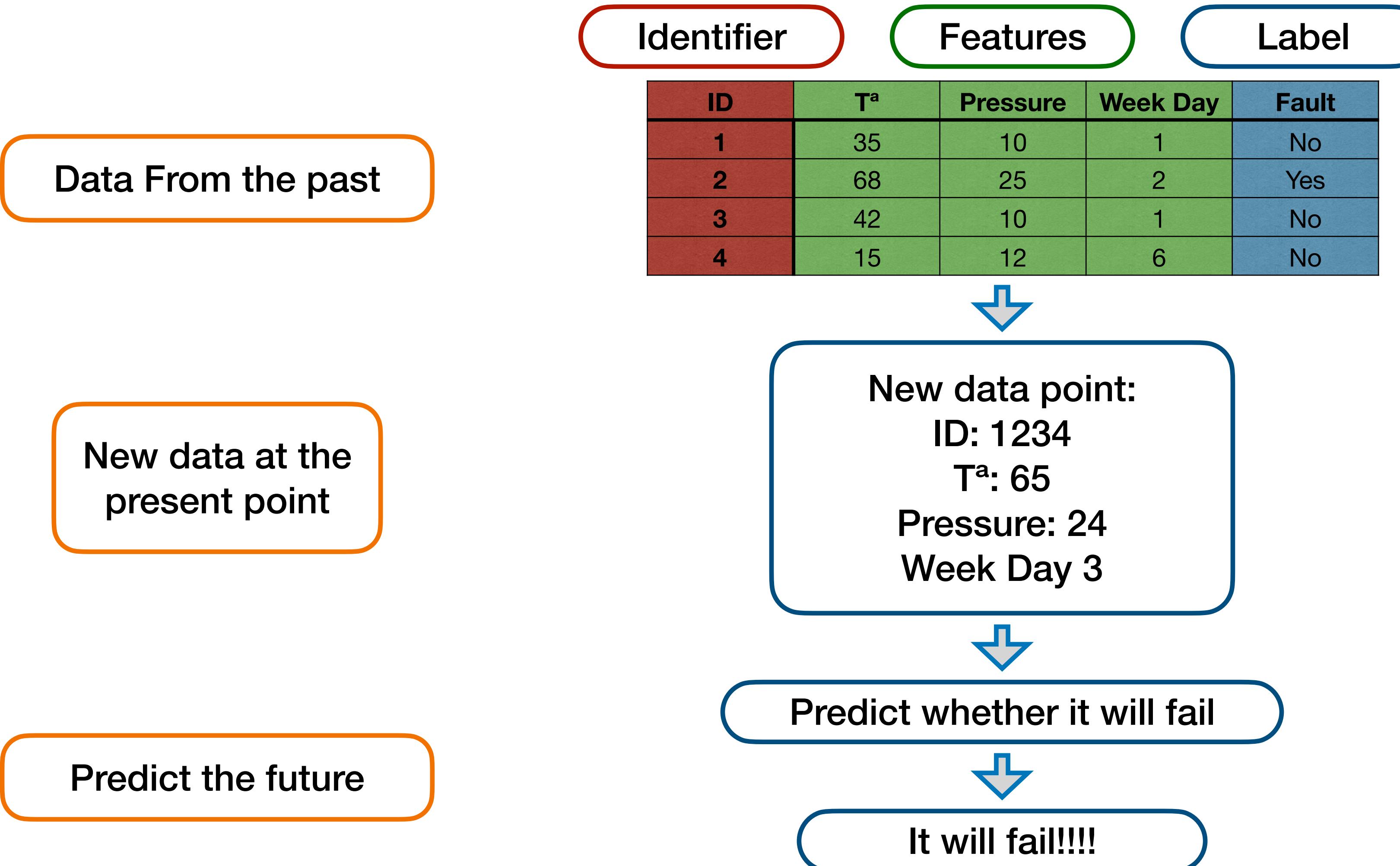
ID	#Bedrooms	#bathrooms	M2	Price(Euro)
1	3	2	120	190.000
2	3	3	150	250.000
3	1	1	60	100.000
4	2	2	80	140.000

Predict House Prices

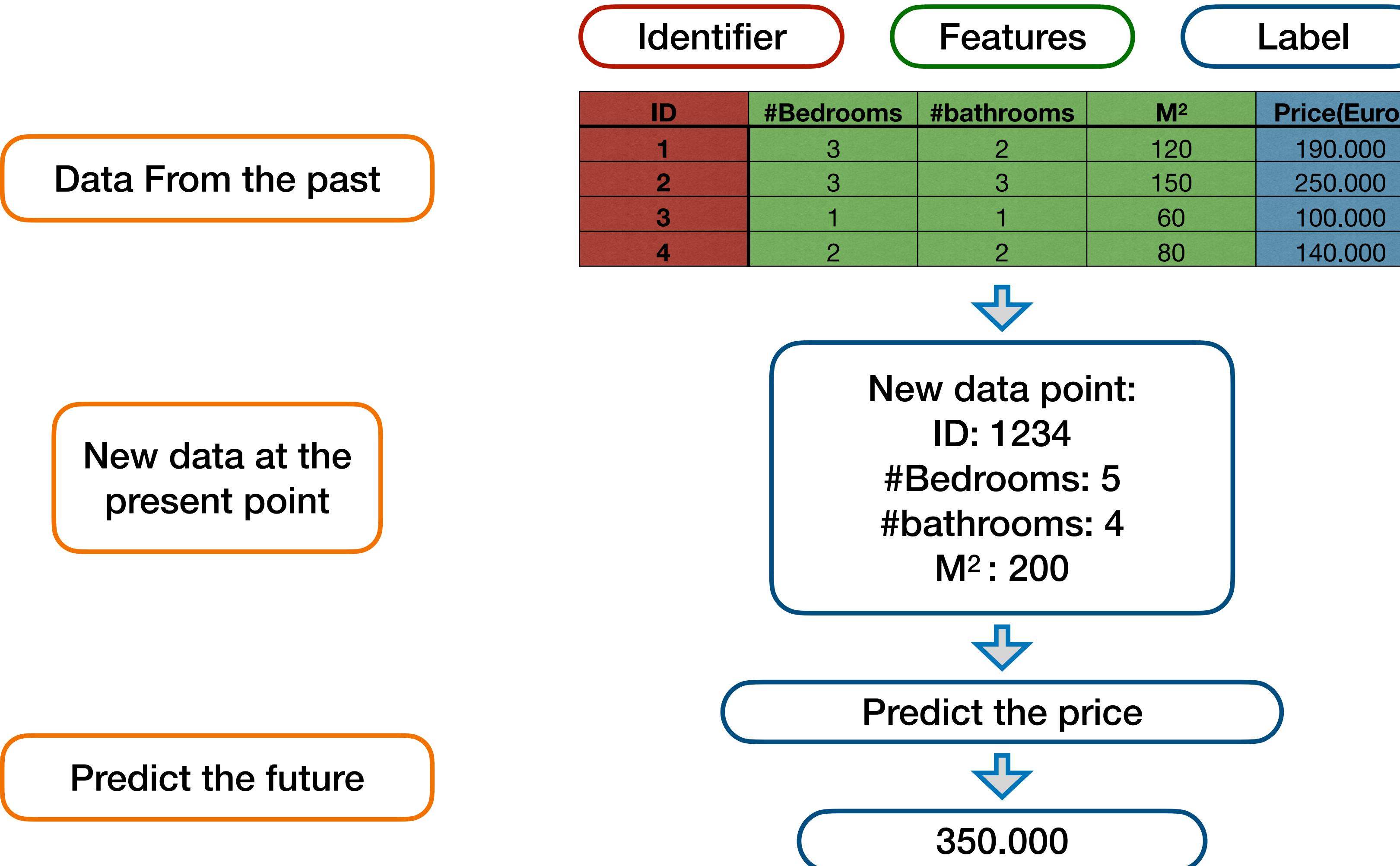
Name	MeanGrades	#PassedSubjects	Dropout
David	4	10	Yes
Evi	8	20	No
Fran	9	25	No
Encarna	7	18	No

Predict Student Dropout

# Learning from the past to predict the future



# Learning from the past to predict the future



# How do we learn from the past? We need something. A model

---

Data From the past

Identifier	Features	Label		
ID	#Bedrooms	#bathrooms	M <sup>2</sup>	Price(Euro)
1	3	2	120	190.000
2	3	3	150	250.000
3	1	1	60	100.000
4	2	2	80	140.000

# How do we learn from the past? We need something. A model

---

Data From the past

Identifier		Features		Label
ID	#Bedrooms	#bathrooms	M <sup>2</sup>	Price(Euro)
1	3	2	120	190.000
2	3	3	150	250.000
3	1	1	60	100.000
4	2	2	80	140.000



Train a model  
using the data  
from the past

Predictive Model  
Black box

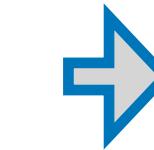
# How do we learn from the past? We need something. A model

Data From the past

Identifier		Features			Label
ID	#Bedrooms	#bathrooms	M <sup>2</sup>	Price(Euro)	
1	3	2	120	190.000	
2	3	3	150	250.000	
3	1	1	60	100.000	
4	2	2	80	140.000	

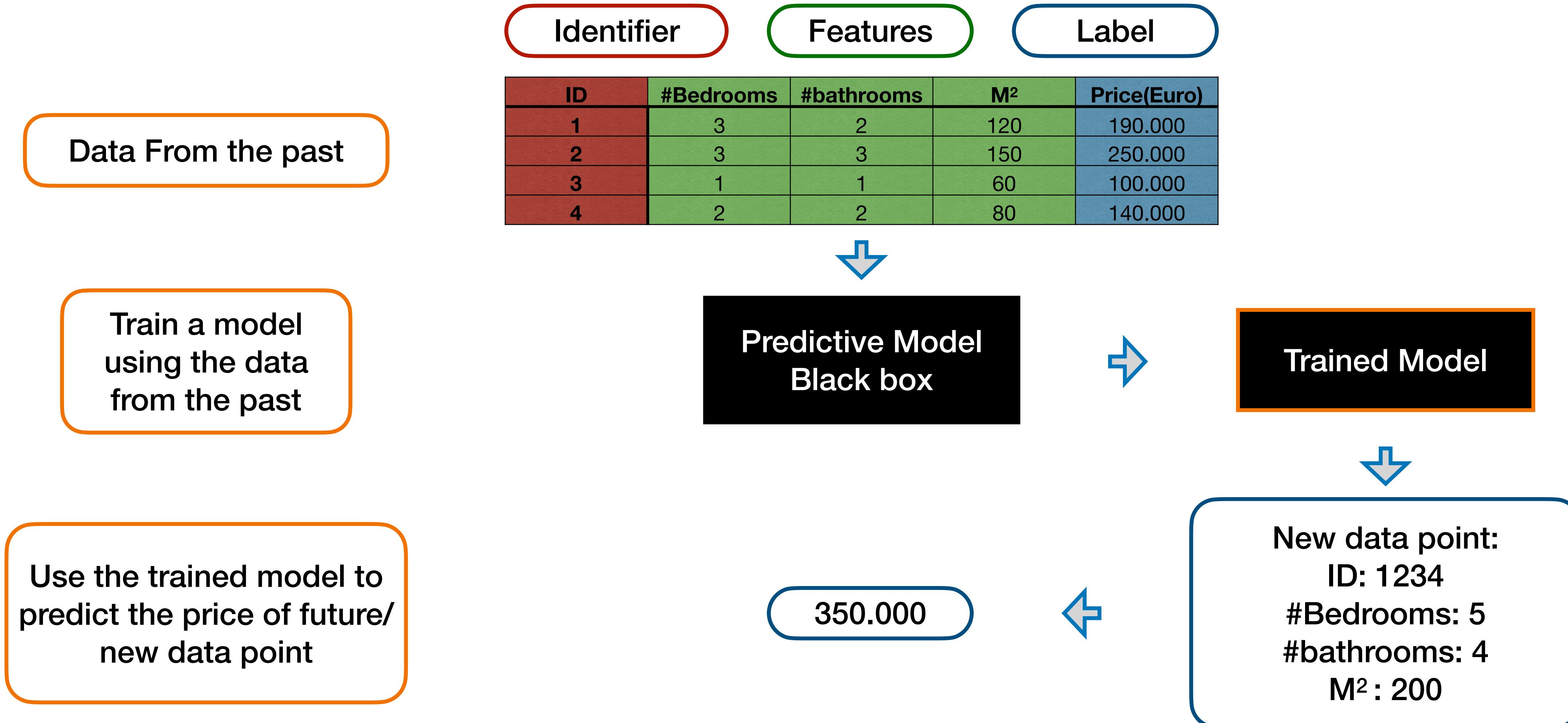
Train a model  
using the data  
from the past

Predictive Model  
Black box



Trained Model

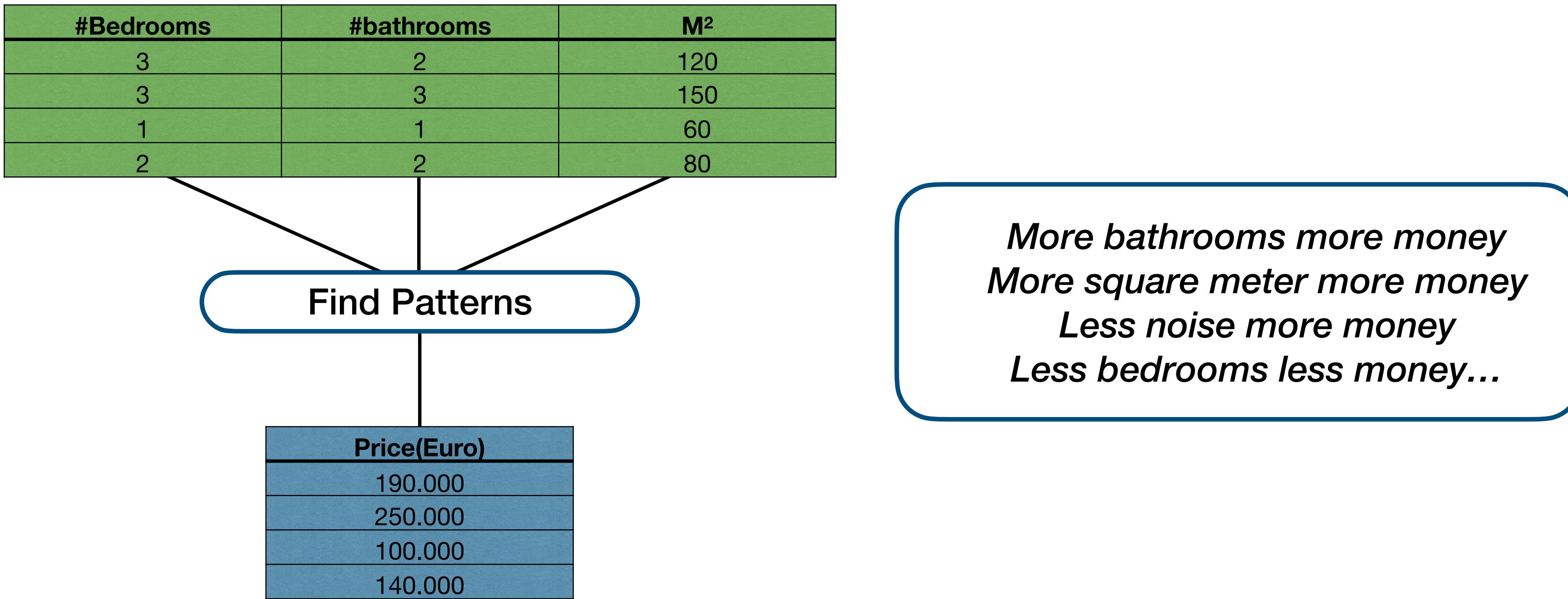
# How do we learn from the past? We need something. A model



# What is a model

---

- Models are programs that can find patterns in the data that is fed to them

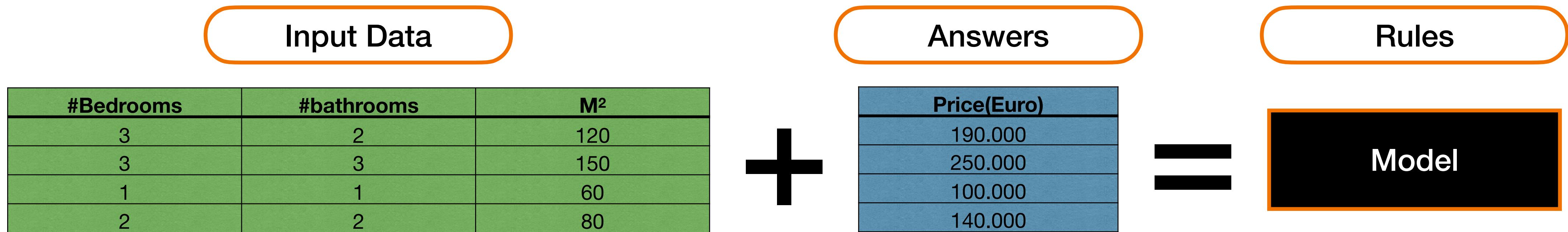


# Topic 3: Supervised vs. Unsupervised. Regression vs. Classification

# AI & Machine Learning

---

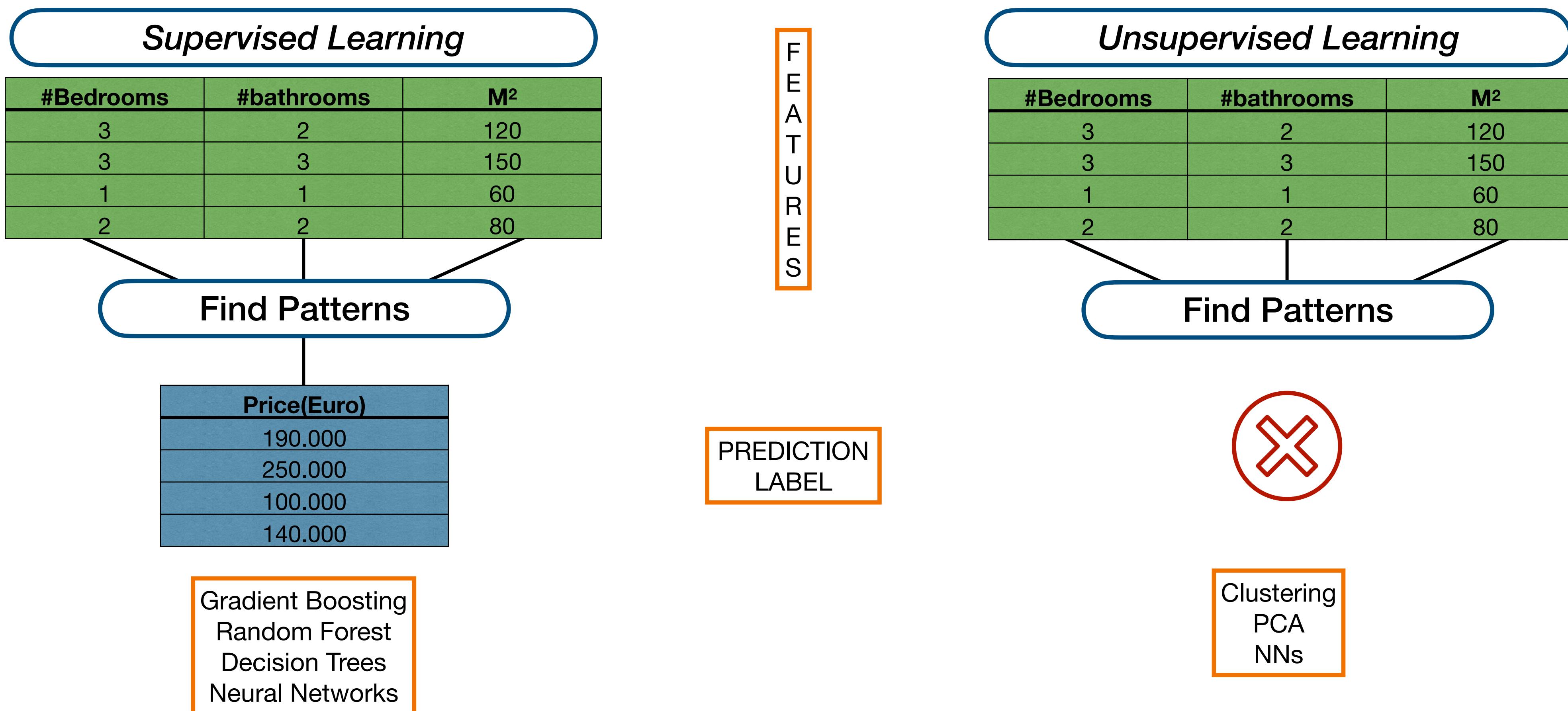
ML: Data (input) + Answers (predictive label) —> Rules (Model)



- Transform data into meaningful outputs  
(Learning from exposure to known examples)

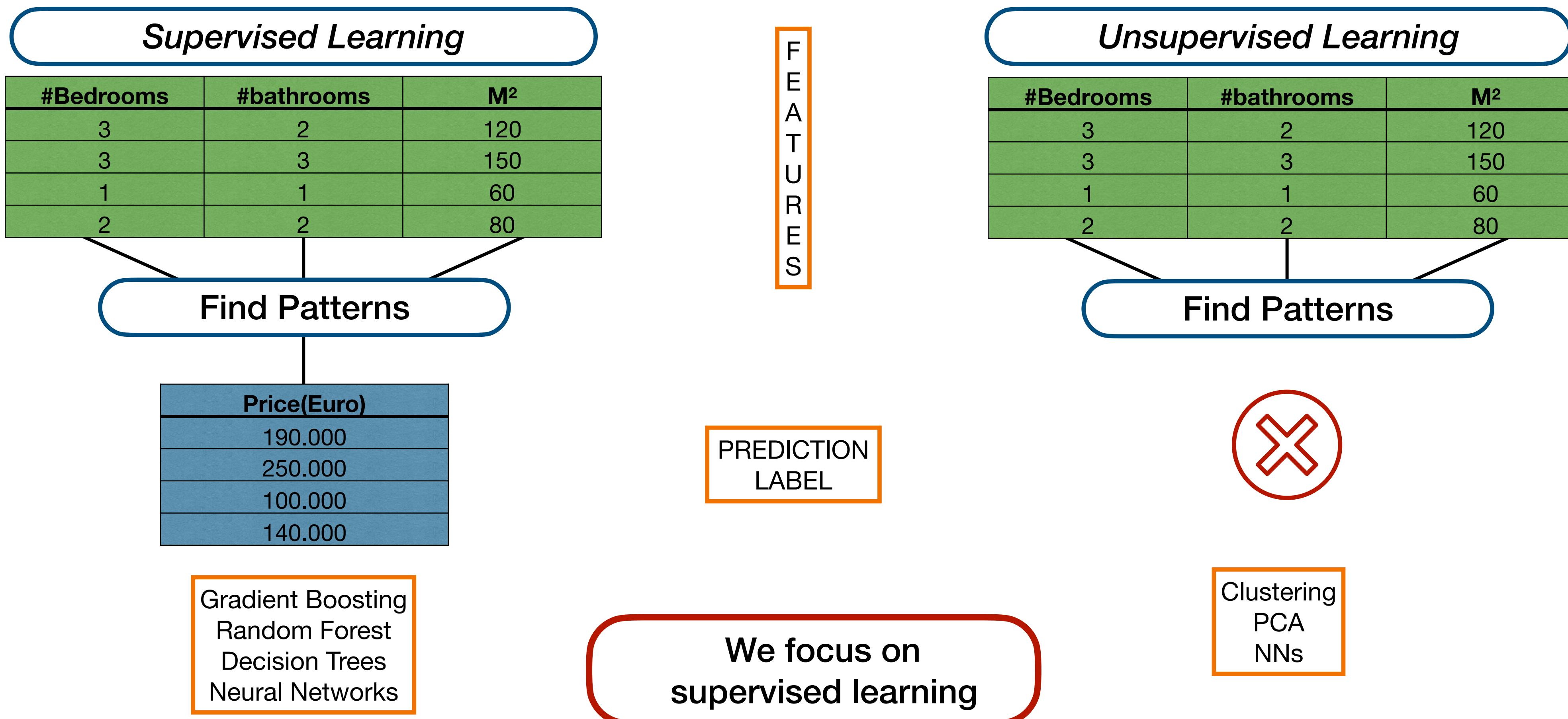
# Supervised vs. Unsupervised learning

- We distinguish between two types of models based on how they function and learn



# Supervised vs. Unsupervised learning

- We distinguish between two types of models based on how they function and learn



# Classification vs. Regression Problems

---

## Classification

ID	T <sup>a</sup>	Pressure	Week Day	Fault
1	35	10	1	No
2	68	25	2	Yes
3	42	10	1	No
4	15	12	6	No

## Regression

ID	#Bedrooms	#bathrooms	M <sup>2</sup>	Price(Euro)
1	3	2	120	190.000
2	3	3	150	250.000
3	1	1	60	100.000
4	2	2	80	140.000

# Classification vs. Regression Problems

## Classification

ID	T <sup>a</sup>	Pressure	Week Day	Fault
1	35	10	1	No
2	68	25	2	Yes
3	42	10	1	No
4	15	12	6	No

## Regression

ID	#Bedrooms	#bathrooms	M <sup>2</sup>	Price(Euro)
1	3	2	120	190.000
2	3	3	150	250.000
3	1	1	60	100.000
4	2	2	80	140.000

*Classification is about predicting a label (Categorical) and regression is about predicting a quantity (Numerical)*

# Topic 4: Arranging data for models

# Computers only understand numbers!

---

Categorical Features		Numerical Features	
Gender	Parent Education	Age	Grade
Male	College	21	8
Female	High School	22	9
Male	None	20	6
Female	College	21	5

Computers need to code the categorical features/columns as numbers. How do we do it?

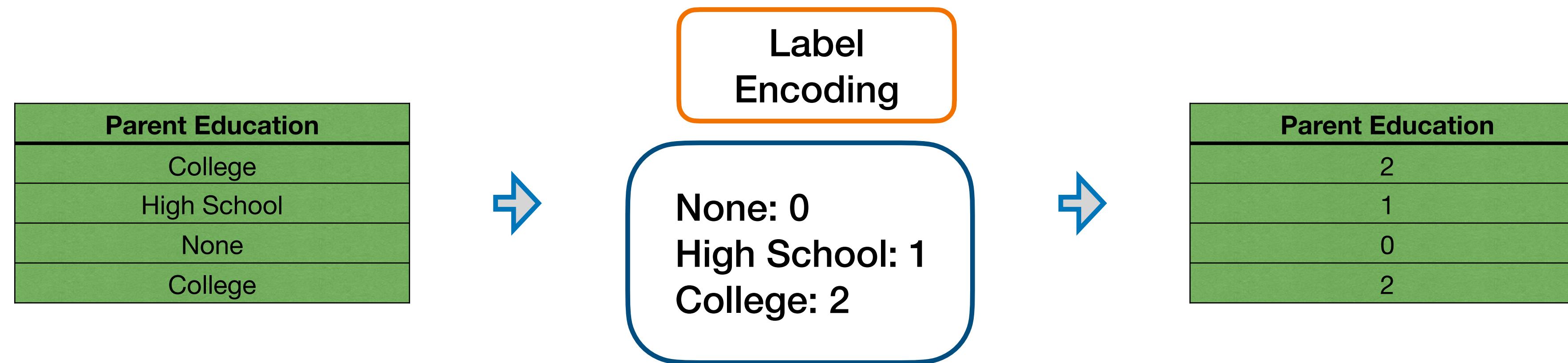
We have two main approaches:

- Label Encoding
- One Hot Encoding

# Label Encoding

---

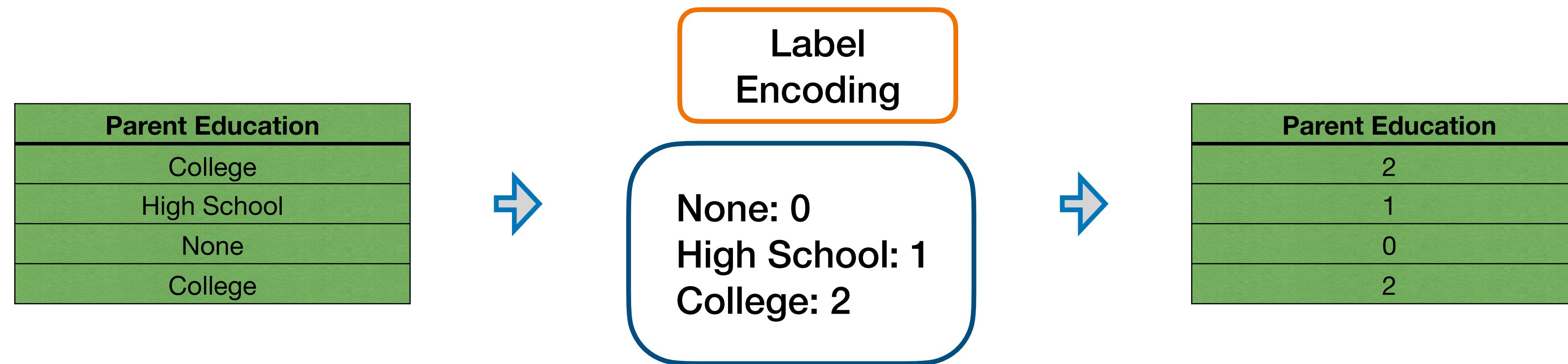
Process of converting the labels into a numeric form so as to convert them into the machine-readable form



# Label Encoding

---

Process of converting the labels into a numeric form so as to convert them into the machine-readable form

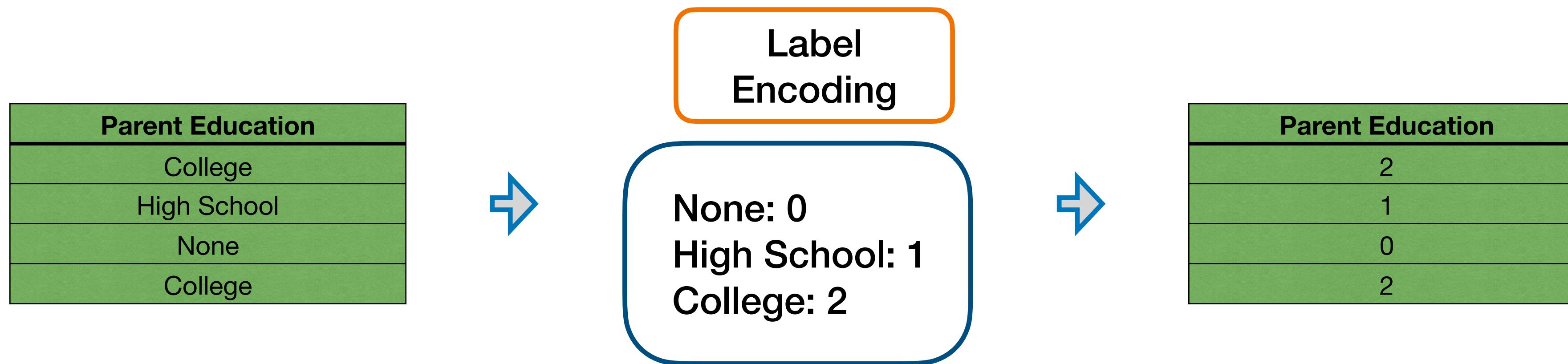


## Limitations / Disadvantages

Numeric values can be “misinterpreted” by the algorithms. For example, the value of 0 is obviously less than the value of 4 but is that so?

# Label Encoding

Process of converting the labels into a numeric form so as to convert them into the machine-readable form



## Limitations / Disadvantages

Numeric values can be “misinterpreted” by the algorithms. For example, the value of 0 is obviously less than the value of 4 but is that so?

## Python Code

```
df["parent_ed_cat_encode"] = df["parental level of education"]\n    .astype('category').cat.codes
```

```
df[['parental level of education', 'parent_ed_cat_encode']].head()
```

parental level of education	parent_ed_cat_encode
high school	2
some college	4
high school	2
associate's degree	0
high school	2

# Exercise

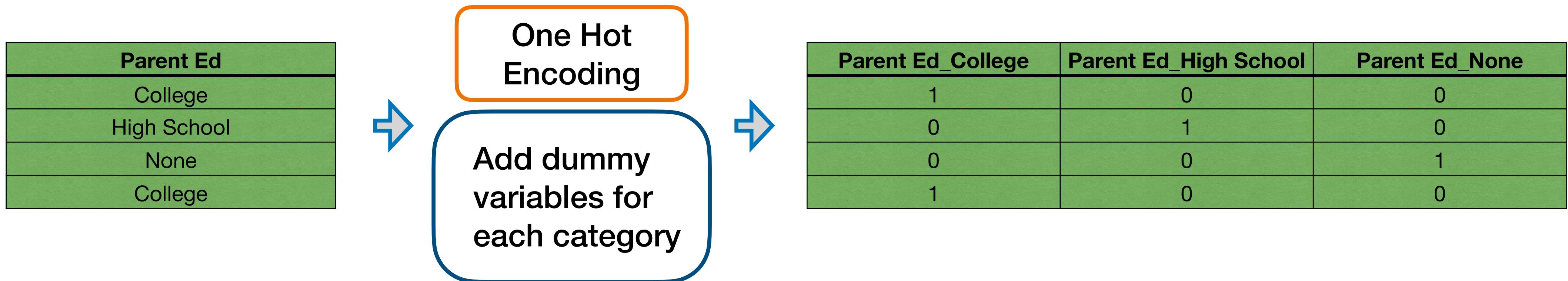
---

- Test level encoding to the column race/ethnicity. Delete the new column when done or do not modify the source dataframe.

# One hot encoding

---

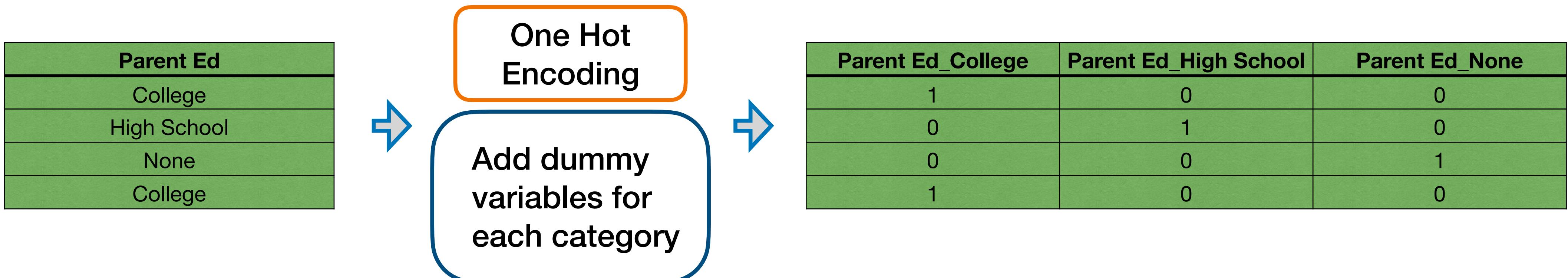
Convert each category value into a new column and assigns a 1 or 0 (True/False) value to the column.



# One hot encoding

---

Convert each category value into a new column and assigns a 1 or 0 (True/False) value to the column.



## Limitations / Disadvantages

It has the benefit of not weighting a value improperly but does have the downside of adding more columns to the data set.

# One hot encoding

Convert each category value into a new column and assigns a 1 or 0 (True/False) value to the column.

Parent Ed
College
High School
None
College

One Hot Encoding

Add dummy variables for each category

Parent Ed_College	Parent Ed_High School	Parent Ed_None
1	0	0
0	1	0
0	0	1
1	0	0

Limitations / Disadvantages

It has the benefit of not weighting a value improperly but does have the downside of adding more columns to the data set.

Python Code

```
pd.get_dummies(df['gender'], columns=['gender']).head()
```

	female	male
0	0	1
1	0	1
2	0	1
3	0	1
4	1	0

# Exercise

---

- Do one-hot encoding of all categorical columns

# Train & Test sets. Why are they necessary?

---

**Overfitting:** The model is not good on never seen data

**Underfitting:** The model is bad everywhere

The reason to have a separate train and test set is to avoid overfitting of the model to the data

# Train & Test sets. Why are they necessary?

---

Overfitting: The model is not good on never seen data

Underfitting: The model is bad everywhere

The reason to have a separate train and test set is to avoid overfitting of the model to the data

Training Set

Gender	Parent Education	Age	Grade
Male	College	21	8
Female	High School	22	9

Test Set

Gender	Parent Education	Age	Grade
Male	None	20	6
Female	College	21	5

To train the model. Can represent up to 90 % of the data

To test how the model generalizes to never seen data

# Topic 4: Intro to ML models in Python

# Our Pandas for ML models is Scikit Learn

---



Scikit learn is an ML library for Python that features various classification, regression and clustering algorithms

It will allow us to build a model and make predictions in 4 lines of code

```
from sklearn import datasets, linear_model
from sklearn.metrics import mean_absolute_error

# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(trainX, trainY)

# Make predictions using the testing set
y_pred = regr.predict(testX)

# The mean absolute error
print("MAE: %.2f" % mean_absolute_error(testY, y_pred))
```

MAE: 4.41

# Documentation

# Datacamp Courses

---

## Python Fundamentals

Are you ready to gain the foundational skills you need to become a Python programmer? In this track, you'll learn the Python basics you need to start on your programming journey, including how to clean real-world data ready for analysis, use data visualization libraries, and even how to write your own Python functions.

Your instructor Hugo will introduce you to how companies worldwide use Python to gain a competitive edge. Through hands-on coding exercises you'll then learn how to store, manipulate, and explore data using NumPy. Then it's time to level-up as you learn how to visualize your data using Matplotlib, manipulate DataFrames and dictionaries using pandas, and write your own functions and list comprehension. Start this track to add these essential Python skills to your data science toolbox.

[Switch Track](#)

Python 15 hours 4 Courses 1 Skill Assessment

## Data Scientist with Python

Gain the career-building Python skills you need to succeed as a data scientist. No prior coding experience required.

In this track, you'll learn how this versatile language allows you to import, clean, manipulate, and visualize data—all integral skills for any aspiring data professional or researcher. Through interactive exercises, you'll get hands-on with some of the most popular Python libraries, including pandas, NumPy, Matplotlib, and many more. You'll then work with real-world datasets to learn the statistical and machine learning techniques you need to train decision trees and use natural language processing (NLP). Start this track, grow your Python skills, and begin your journey to becoming a confident data scientist.

[Resume Track](#)

Python 88 hours 23 Courses 6 Projects 3 Skill Assessments

# Free Books to learn Python & data analytics

---

