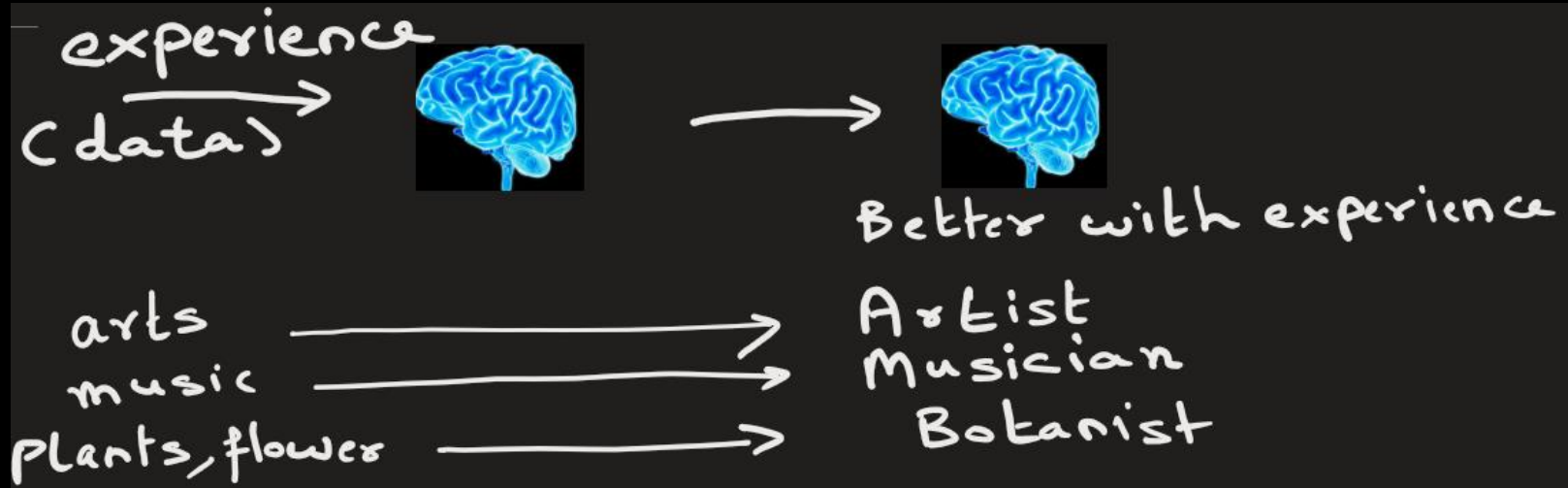


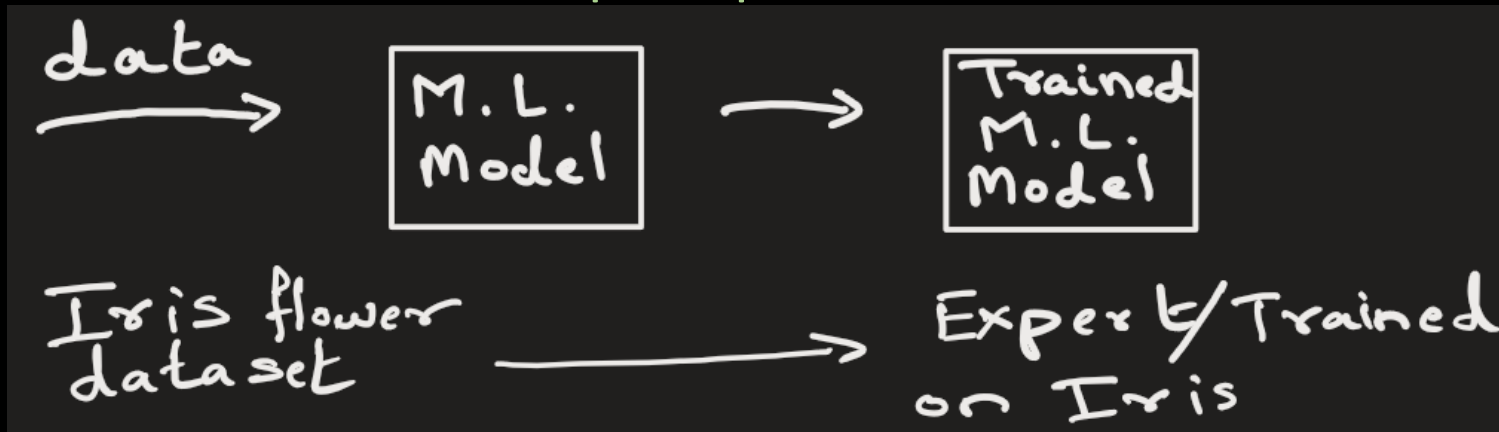
Machine Learning – General Idea

Human brain learns by experience.

This experience comes in the form of data. The brain becomes better with more data.



ML Model works on the same principle.





Task: Make a ML model that would become expert on Iris flower. You are given data of Iris flower.

The ML model should be able to recognize the type of flower (Setosa, Versicolor or Virginica) based on input: sepal and petal dimensions

<u>SepalLength</u>	<u>SepalWidth</u>	<u>PetalLength</u>	<u>PetalWidth</u>	<u>Species</u>
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
7	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4	1.3	Iris-versicolor
6.3	3.3	6	2.5	Iris-virginica
5.8	2.7	5.1	1.9	Iris-virginica
7.1	3	5.9	2.1	Iris-virginica



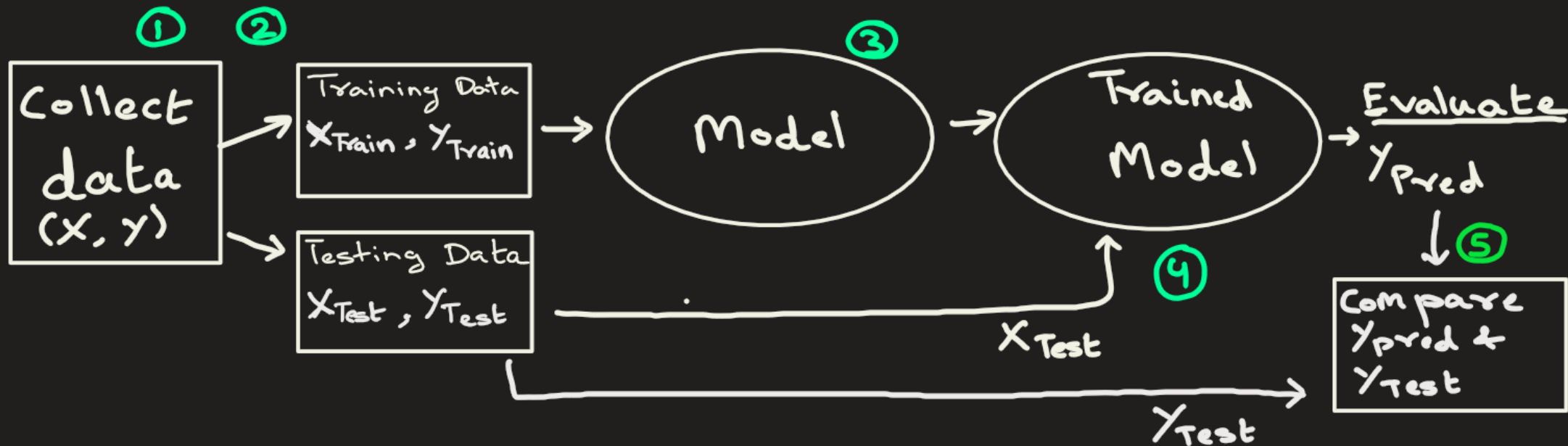
Iris Versicolor



Iris Setosa



Iris Virginica



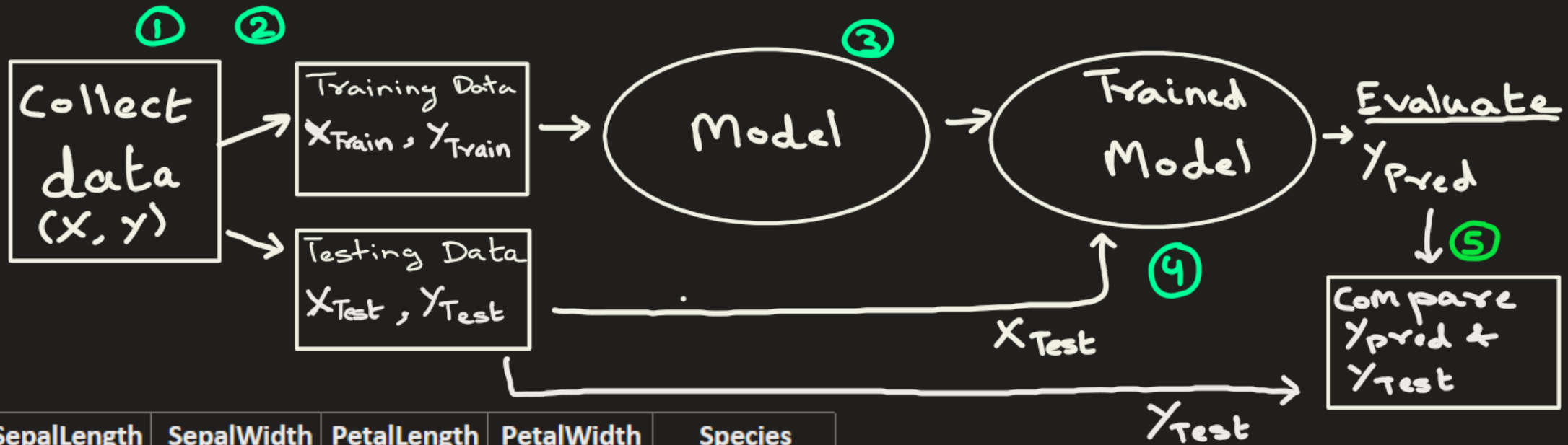
```
import pandas as pd
from sklearn.datasets import load_iris

# 1. LOAD DATASET
dataset = load_iris()

X = pd.DataFrame(dataset.data, columns=dataset.feature_names)
y = dataset.target

print("Dataset shape:", X.shape)
print("Classes:", dataset.target_names)
```


```
Dataset shape: (150, 4)
Classes: ['setosa' 'versicolor' 'virginica']
```




<u>SepalLength</u>	<u>SepalWidth</u>	<u>PetalLength</u>	<u>PetalWidth</u>	<u>Species</u>
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
7	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4	1.3	Iris-versicolor
6.3	3.3	6	2.5	Iris-virginica
5.8	2.7	5.1	1.9	Iris-virginica
7.1	3	5.9	2.1	Iris-virginica

Following is a sample taken from Iris flower dataset.

- This dataset has total of 150 records
- It contains records for 3 types of flowers: Setosa, Versicolor and Virginica
- Each type of flower has 50 records.



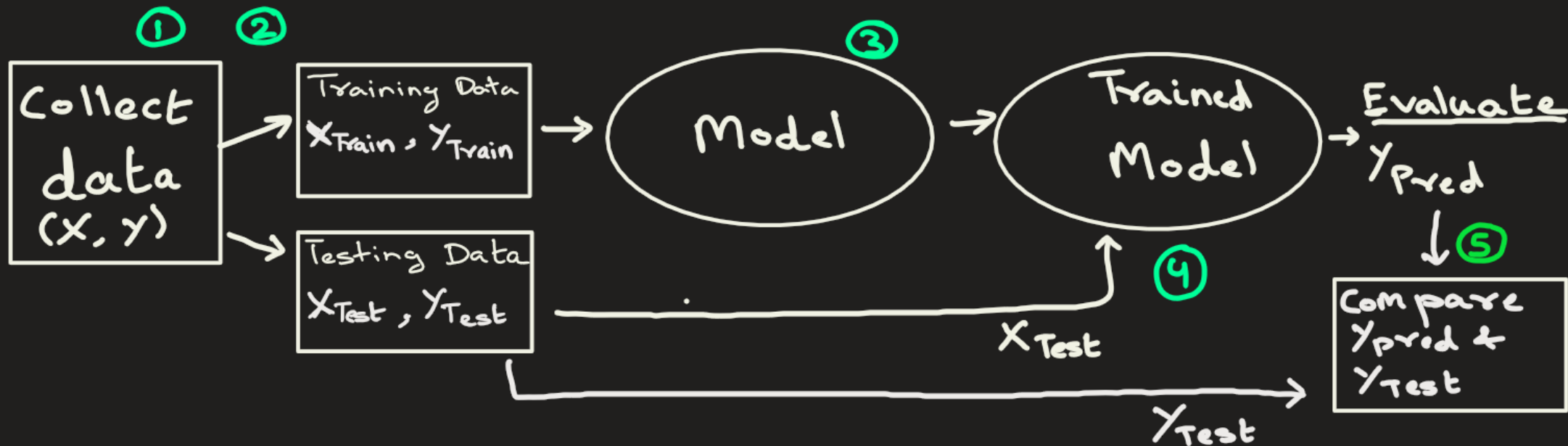
X is called **features** (independent)
y is called **target/response/label** (dependent)
y is function of X.



X

Y

<u>SepalLength</u>	<u>SepalWidth</u>	<u>PetalLength</u>	<u>PetalWidth</u>	<u>Species</u>
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
7	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4	1.3	Iris-versicolor
6.3	3.3	6	2.5	Iris-virginica
5.8	2.7	5.1	1.9	Iris-virginica
7.1	3	5.9	2.1	Iris-virginica



2. TRAIN-TEST SPLIT

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,)
```

```
print("shape of X_train:", X_train.shape)
```

```
print("shape of X_test:", X_test.shape)
```

```
shape of X_train: (120, 4)
```

```
shape of X_test: (30, 4)
```




ML Overview



X and y are split into training and testing part.
There would be total 4 parts now.

X

X_{Train}

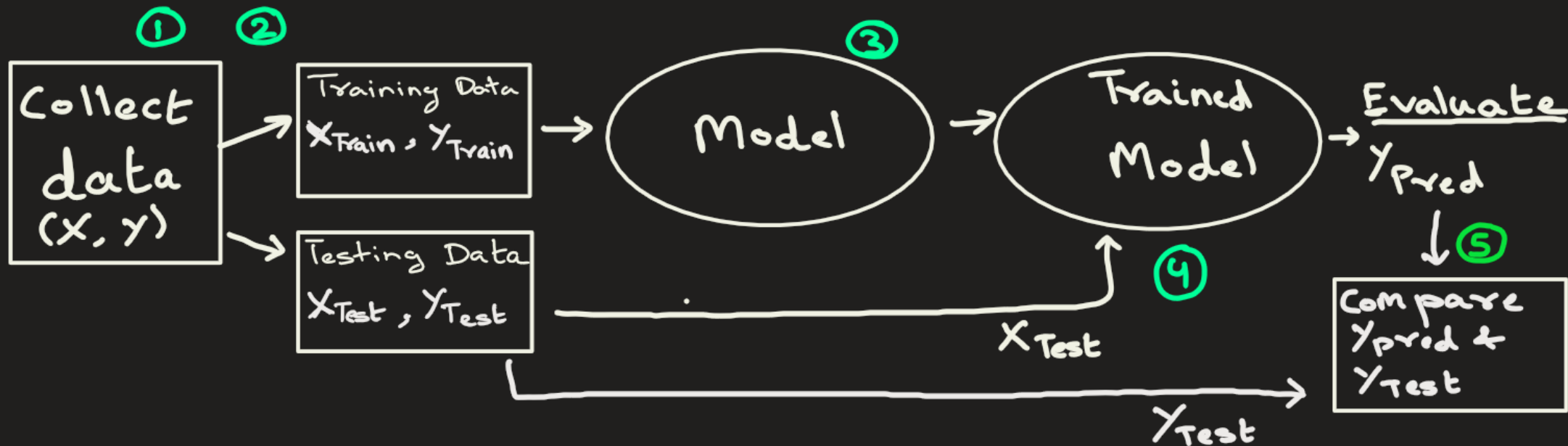
SepalLength	SepalWidth	PetalLength	PetalWidth
5.1	3.5	1.4	0.2
4.9	3	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
7	3.2	4.7	1.4
6.4	3.2	4.5	1.5
6.9	3.1	4.9	1.5
5.5	2.3	4	1.3

Y_{Train}

X_{Test}

6.3	3.3	6	2.5
5.8	2.7	5.1	1.9
7.1	3	5.9	2.1

Y_{Test}



3. TRAIN MODEL

```
from sklearn.ensemble import RandomForestClassifier
```

```
model = RandomForestClassifier(  
    n_estimators=200,  
    random_state=42  
)
```

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



RandomForestClassifier

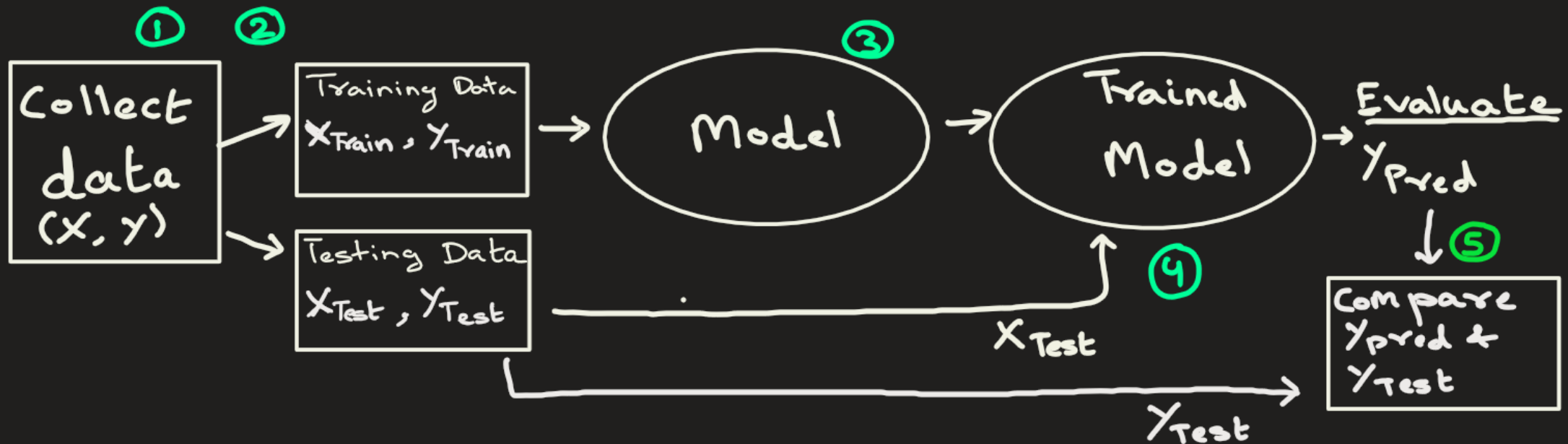


RandomForestClassifier(n_estimators=200, random_state=42)

Training data is fed to the ML Model.
It may take some time for model to learn/train.

The diagram illustrates the split of the Iris dataset into training and testing sets. A large green 'X' is positioned above the training data section, and a large yellow 'Y' is positioned above the testing data section. The training data is enclosed in a green box and labeled X_{Train} on the left and Y_{Train} on the right. The testing data is enclosed in a yellow box and labeled X_{Test} on the left and Y_{Test} on the right. Handwritten notes on the right side indicate that the training data is $(X_{Train} + Y_{Train})$ and the testing data is $(X_{Test} + Y_{Test})$.

SepalLength	SepalWidth	PetalLength	PetalWidth	Species
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
7	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4	1.3	Iris-versicolor
6.3	3.3	6	2.5	Iris-virginica
5.8	2.7	5.1	1.9	Iris-virginica
7.1	3	5.9	2.1	Iris-virginica



```
#      sepal length, sepal width, petal length, petal width
sample = [[5.9, 3.0, 5.1, 1.8],]
sample_df = pd.DataFrame(sample, columns=dataset.feature_names)
```

```
prediction = model.predict(sample_df)
print("Prediction:", prediction)
print("Classes:", dataset.target_names)
```

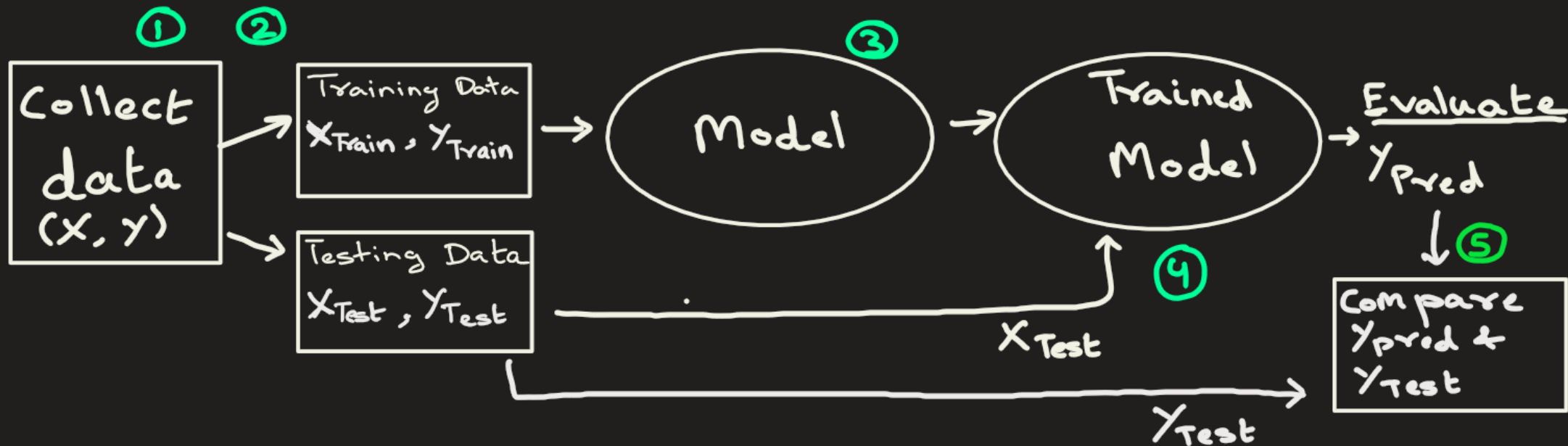
```
Prediction: [2]
Classes: ['setosa' 'versicolor' 'virginica']
```



Iris Versicolor

Iris Setosa

Iris Virginica



4. EVALUATE MODEL

```
y_pred = model.predict(X_test)
```

5. Check accuracy

```
from sklearn.metrics import accuracy_score
```

```
accuracy = accuracy_score(y_test, y_pred)
```

```
print("Model Accuracy:", accuracy)
```

Model Accuracy: 0.9

Evaluate

- X_{test} is fed to the model. Model makes prediction y_{predict} .
- Compare y_{predict} with y_{test} .

Diagram illustrating the data split for evaluation:

Training Data ($X_{\text{Train}} + y_{\text{Train}}$)

Testing Data ($X_{\text{Test}} + y_{\text{Test}}$)

SepalLength	SepalWidth	PetalLength	PetalWidth	Species
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
7	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4	1.3	Iris-versicolor
6.3	3.3	6	2.5	Iris-virginica
5.8	2.7	5.1	1.9	Iris-virginica
7.1	3	5.9	2.1	Iris-virginica



ML Overview

