Explainability metrics

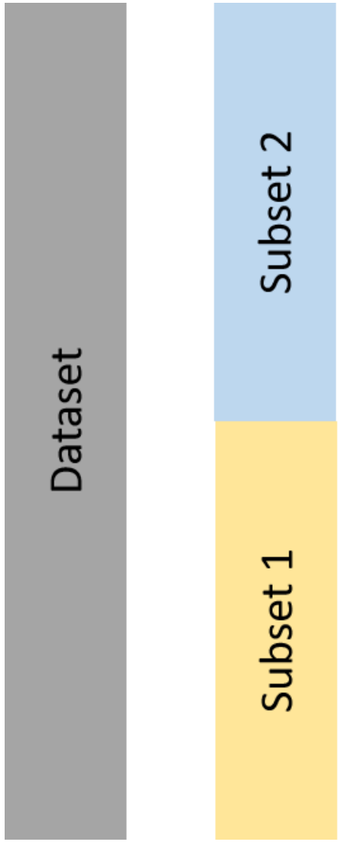
EXPLAINABLE AI IN PYTHON

Fouad Trad
Machine Learning Engineer

datacamp

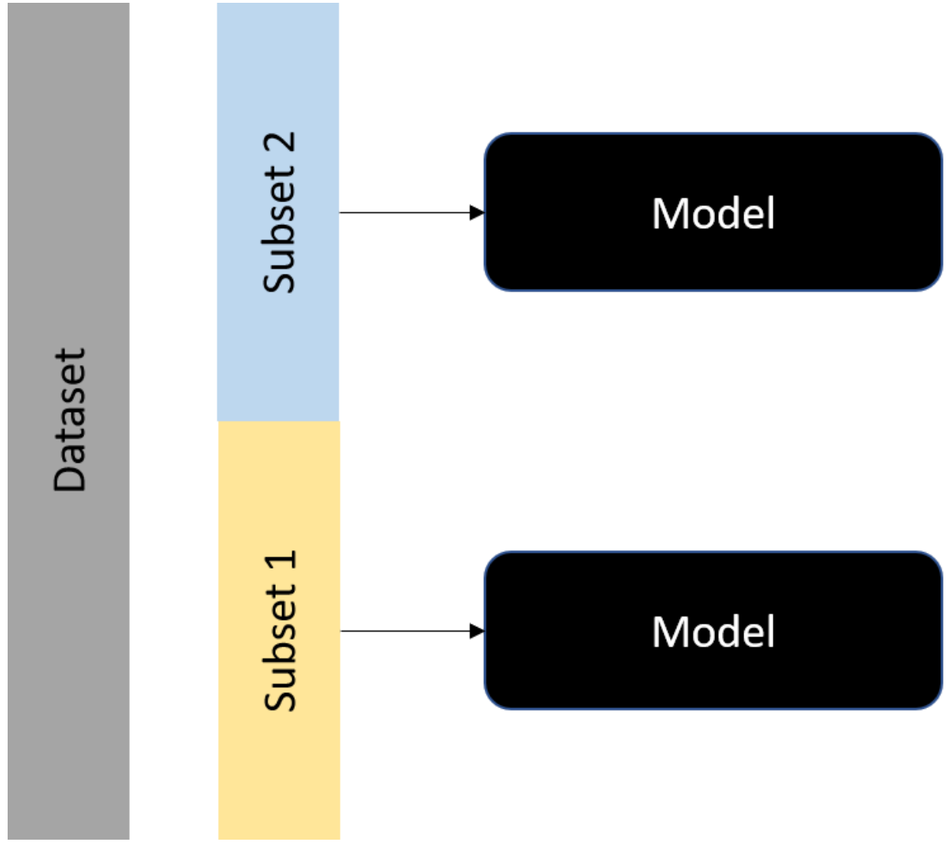
Consistency

- Assesses stability of explanations when model trained on different subsets
- Low consistency → no robust explanations



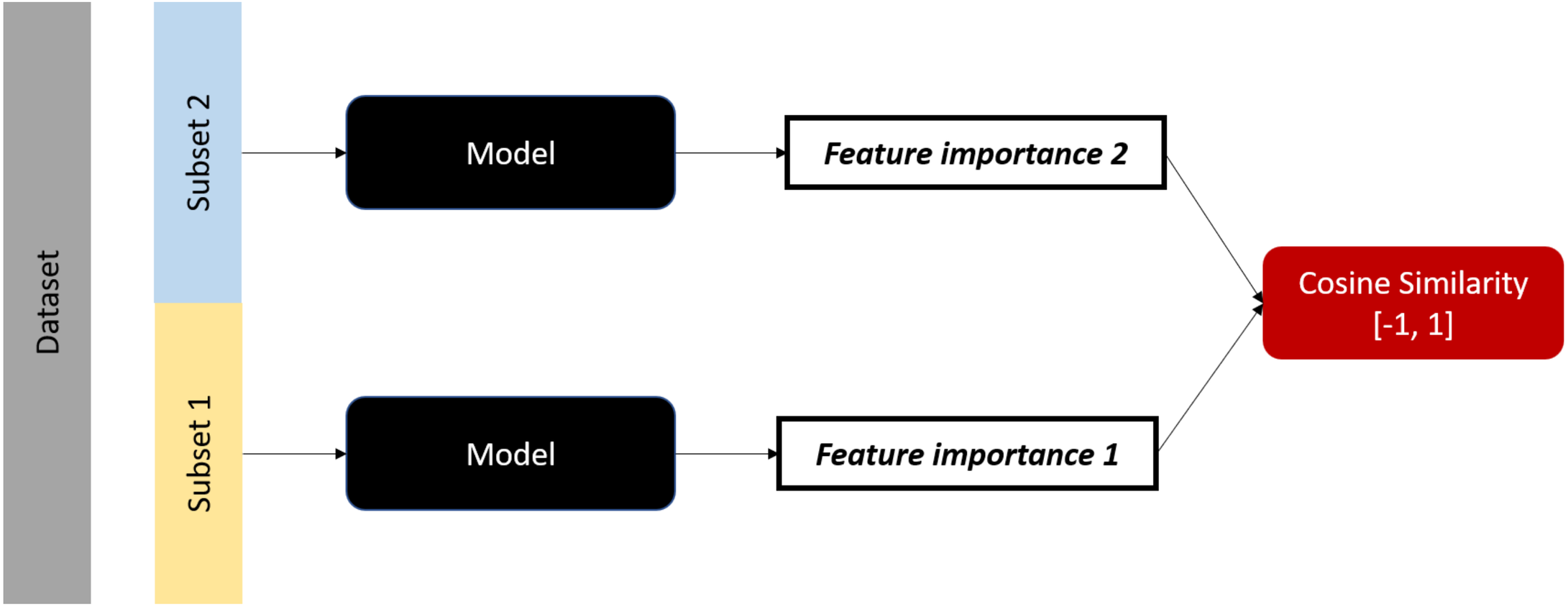
Consistency

- Assesses stability of explanations when model trained on different subsets
- Low consistency → no robust explanations

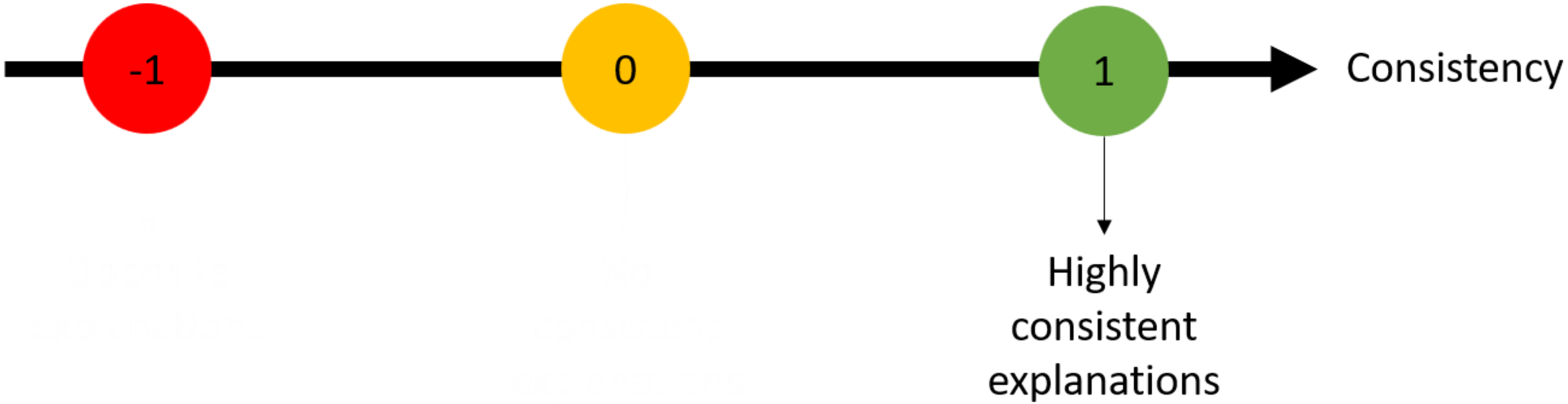


Consistency

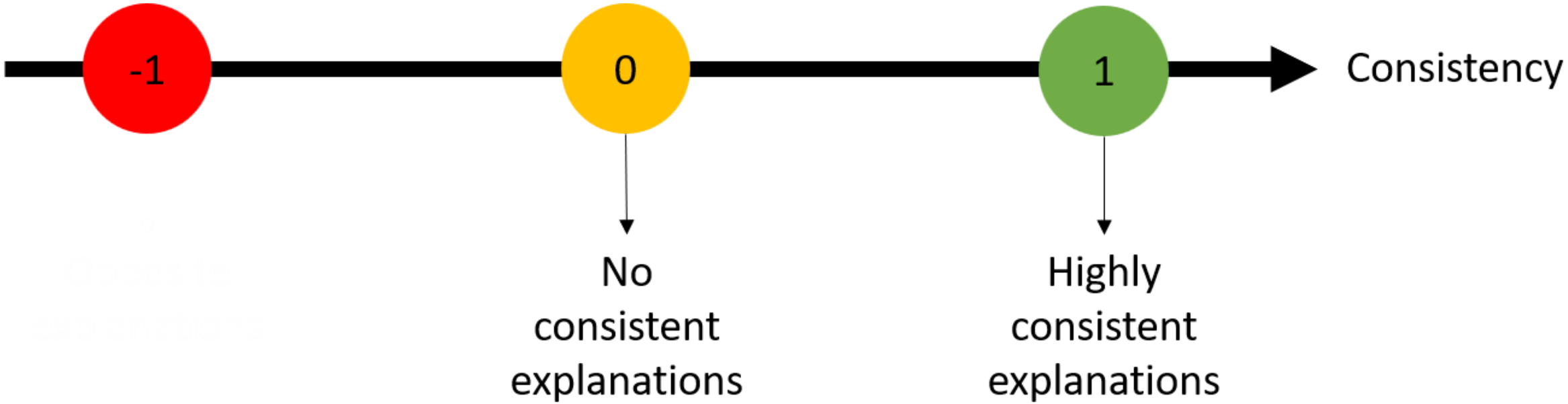
- Assesses stability of explanations when model trained on different subsets
- Low consistency → no robust explanations



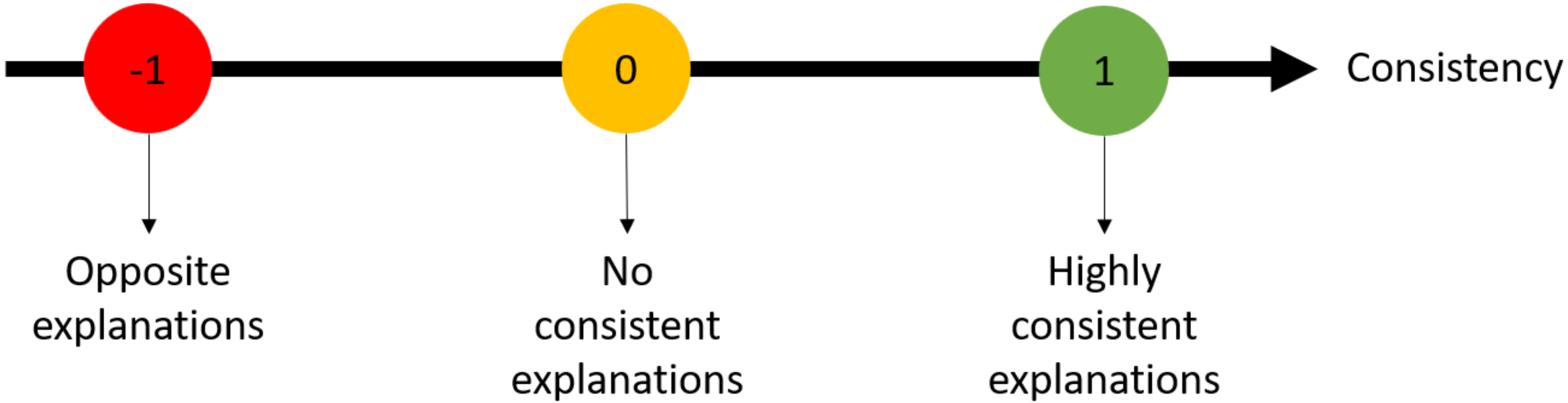
Cosine similarity to measure consistency



Cosine similarity to measure consistency



Cosine similarity to measure consistency



Admissions dataset

GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Chance of Admit
337	118	4	4.5	4.5	9.65	0.92
324	107	4	4	4.5	8.87	0.76
316	104	3	3	3.5	8	0.72
322	110	3	3.5	2.5	8.67	0.8
314	103	2	2	3	8.21	0.45

- `X1` , `y1` : first part of the dataset
- `X2` , `y2` : second part of the dataset
- `model1` , `model2` : random forest regressors

Computing consistency

```
from sklearn.metrics.pairwise import cosine_similarity

explainer1 = shap.TreeExplainer(model1)
explainer2 = shap.TreeExplainer(model2)

shap_values1 = explainer1.shap_values(X1)
shap_values2 = explainer2.shap_values(X2)

feature_importance1 = np.mean(np.abs(shap_values1), axis=0)
feature_importance2 = np.mean(np.abs(shap_values2), axis=0)

consistency = cosine_similarity([feature_importance1], [feature_importance2])
print("Consistency between SHAP values:", consistency)
```

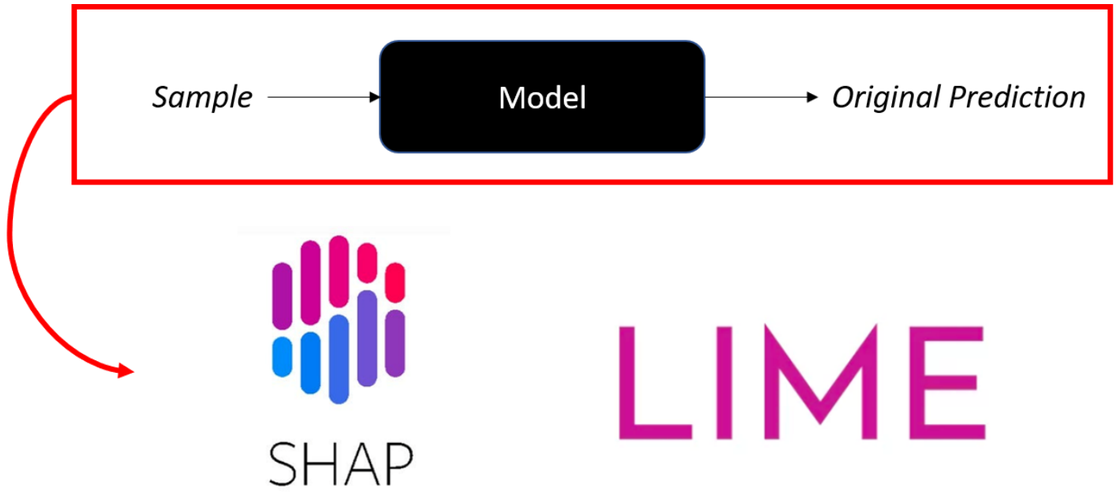
```
Consistency between SHAP values: [[0.99706516]]
```

Faithfulness

- Evaluates if important features influence model's predictions
- Low faithfulness → misleads trust in model reasoning
- Useful in sensitive applications

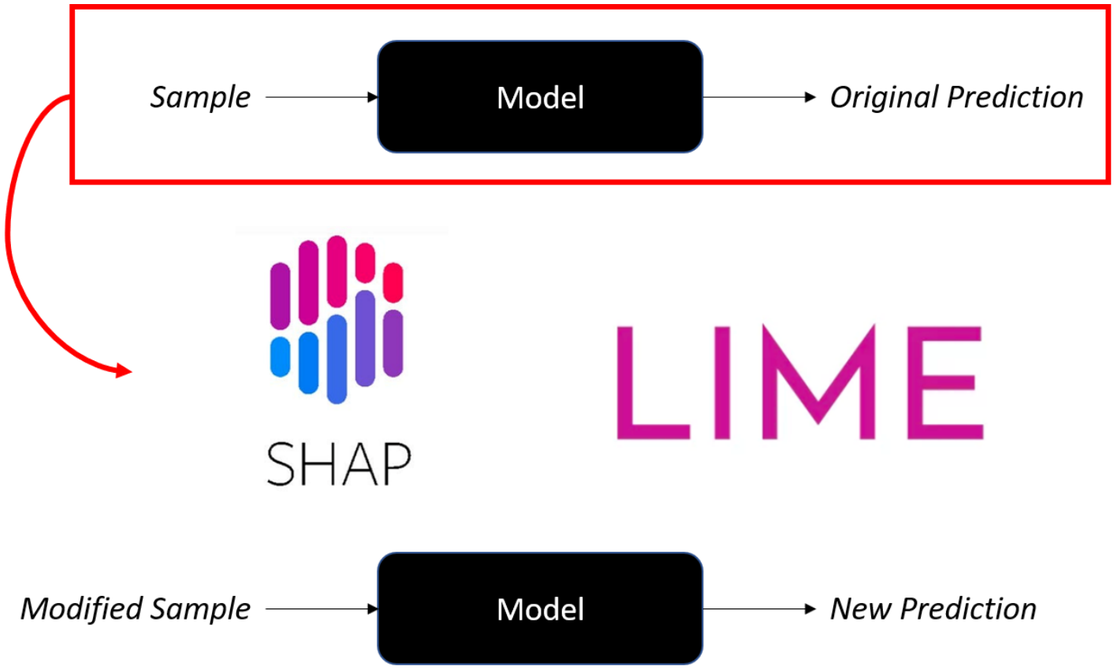
Faithfulness

- Evaluates if important features influence model's predictions
- Low faithfulness → misleads trust in model reasoning
- Useful in sensitive applications



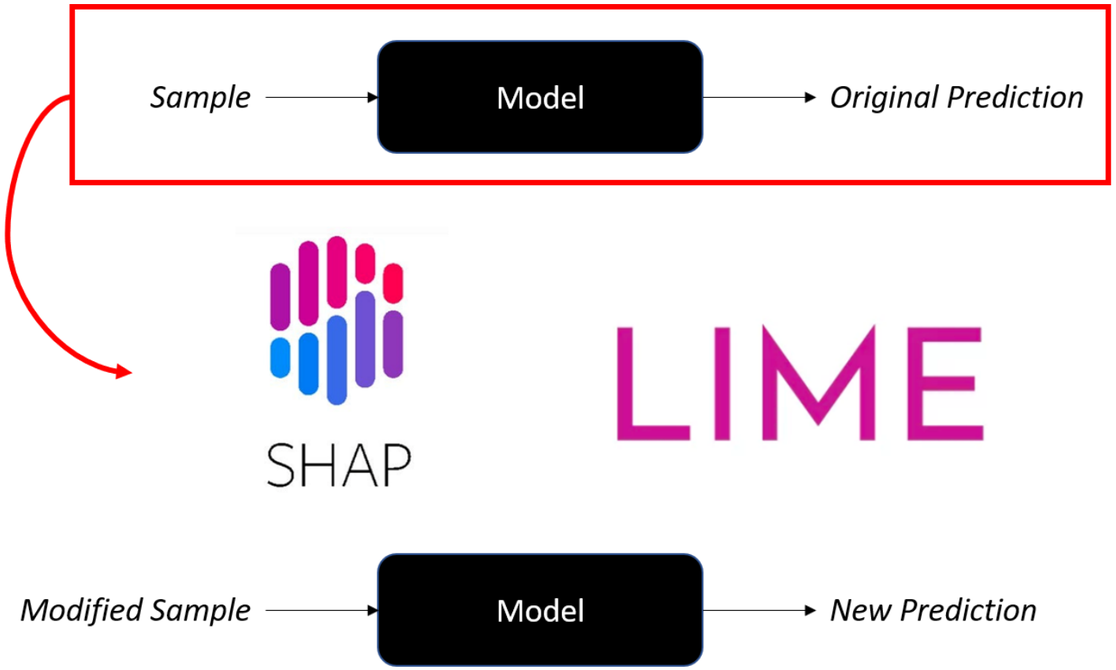
Faithfulness

- Evaluates if important features influence model's predictions
- Low faithfulness → misleads trust in model reasoning
- Useful in sensitive applications



Faithfulness

- Evaluates if important features influence model's predictions
- Low faithfulness → misleads trust in model reasoning
- Useful in sensitive applications

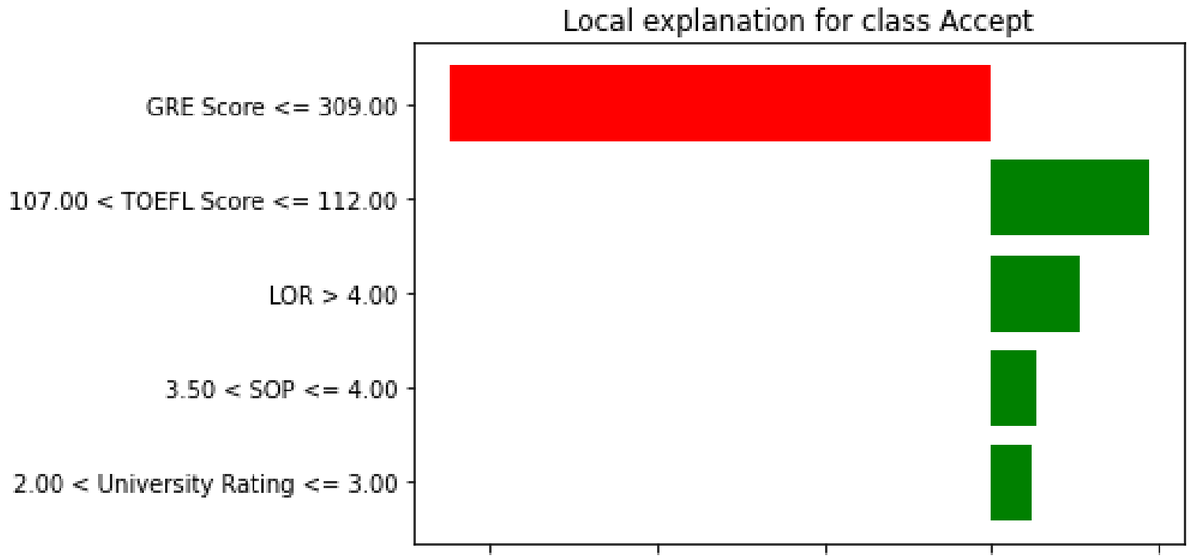


$$\boldsymbol{Faithfulness = Abs(New\ prediction\ - Original\ prediction)}$$

Computing faithfulness

```
X_instance = X_test.iloc[[0]]
original_prediction = model.predict_proba(X_instance)[0, 1]
print(f"Original prediction: {original_prediction}")
```

```
Original prediction: 0.43
```



Computing faithfulness

```
X_instance['GRE Score'] = 310

new_prediction = model.predict_proba(X_instance)[0, 1]
print(f"Prediction after perturbing {important_feature}: {new_prediction}")

faithfulness_score = np.abs(original_prediction - new_prediction)
print(f"Local Faithfulness Score: {faithfulness_score}")
```

```
Prediction after perturbing GRE Score: 0.77
Local Faithfulness Score: 0.34
```

Let's practice!

EXPLAINABLE AI IN PYTHON

Explaining unsupervised models

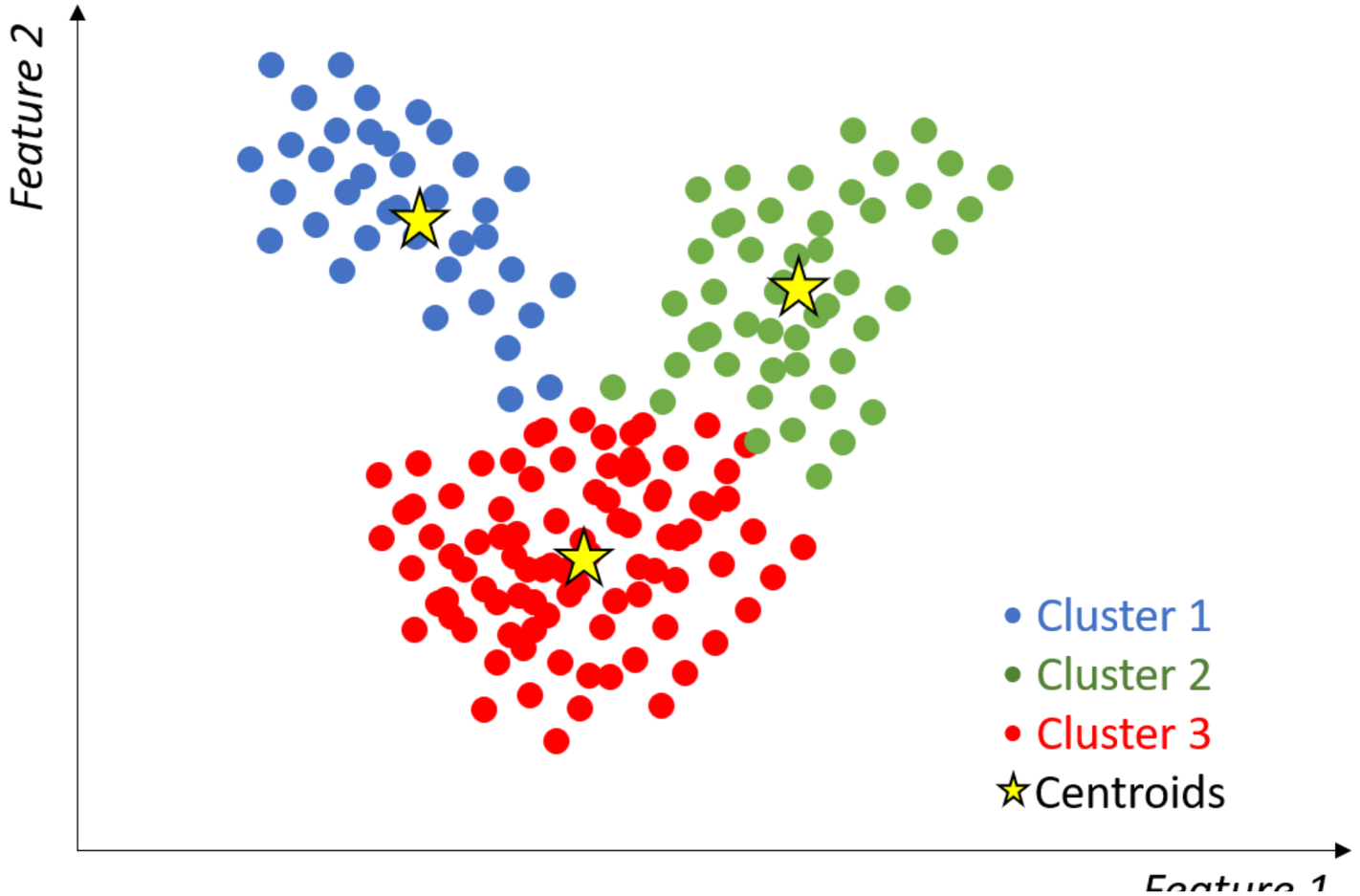
EXPLAINABLE AI IN PYTHON

Fouad Trad
Machine Learning Engineer

datacamp

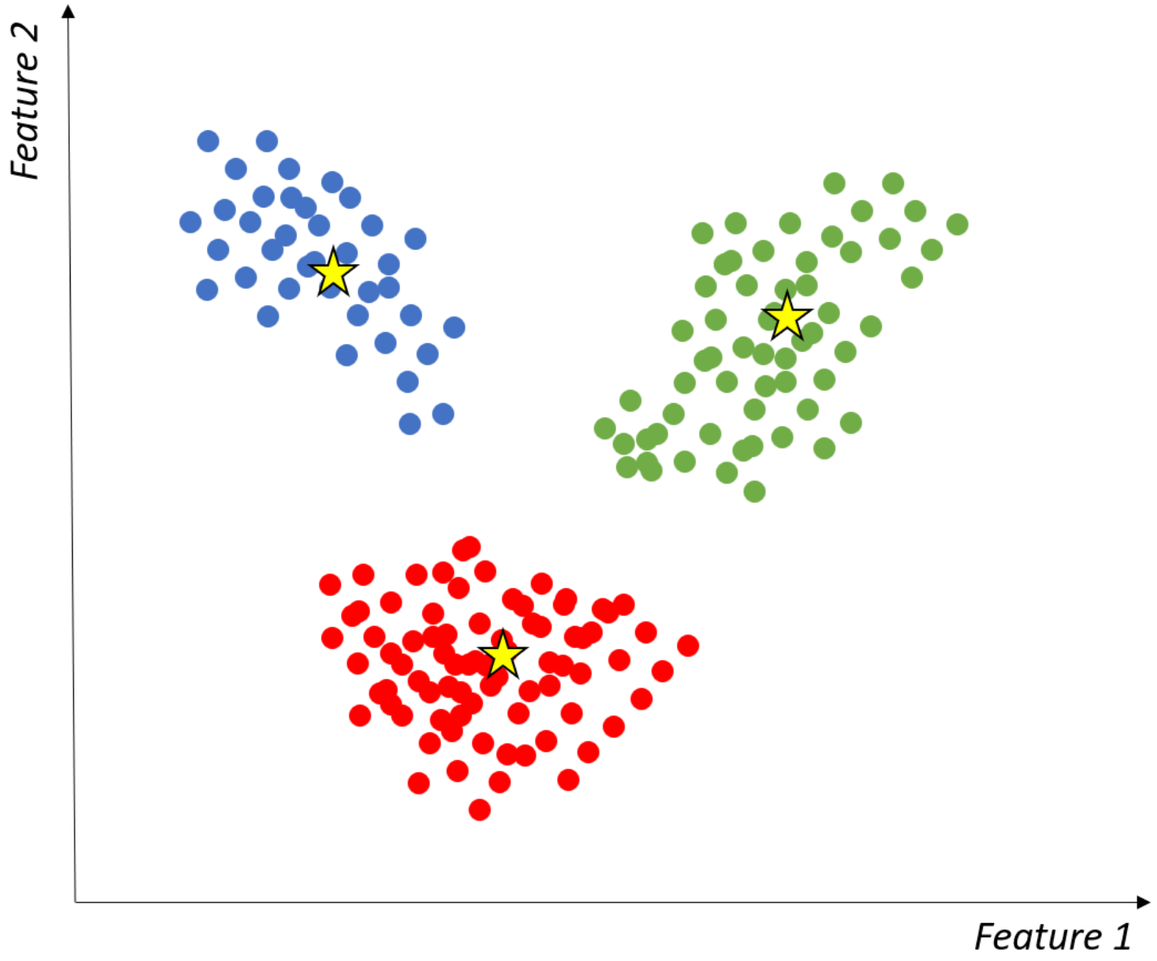
Clustering

Group similar data points without pre-defined labels



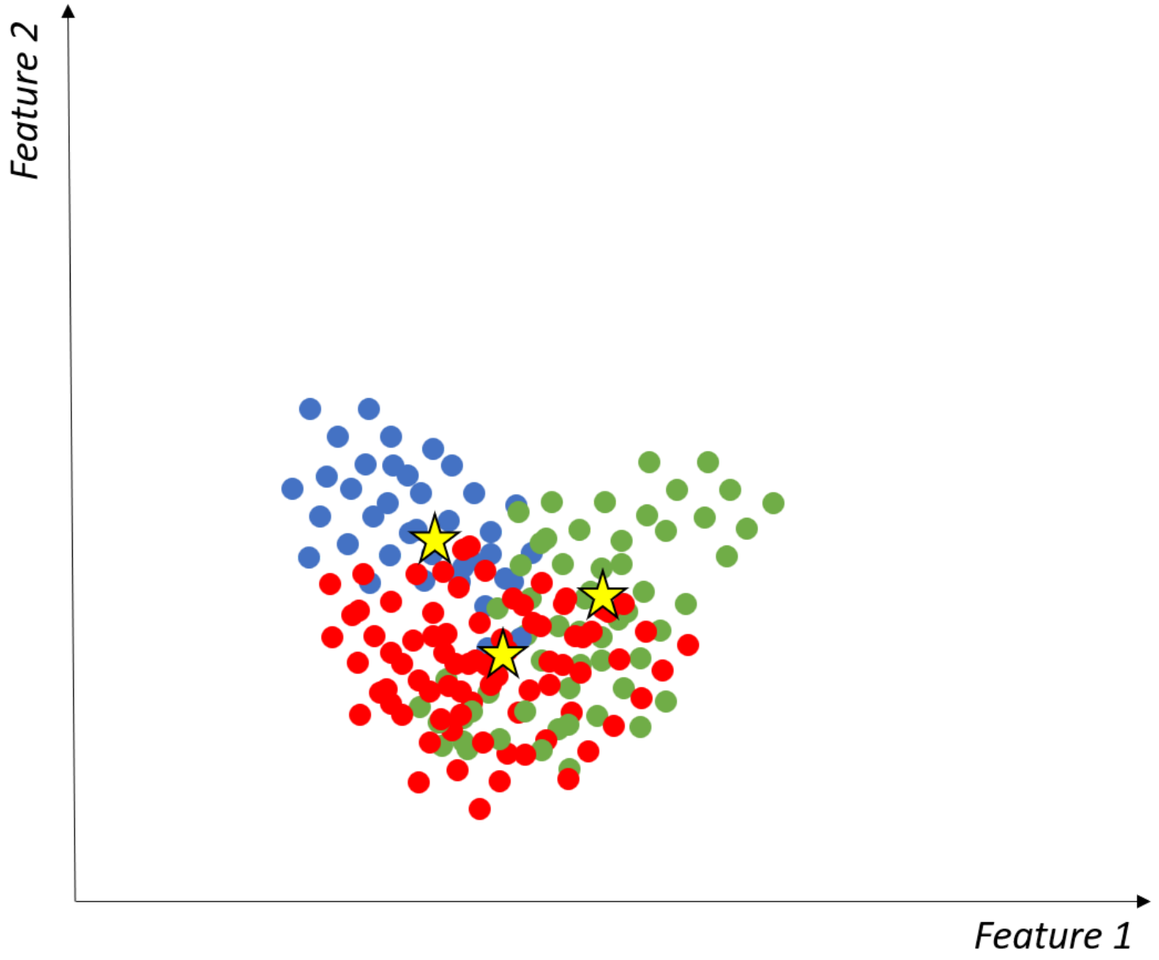
Silhouette score

- Measures clustering's quality
- Ranges from -1 to 1
  - 1 → well-separated clusters



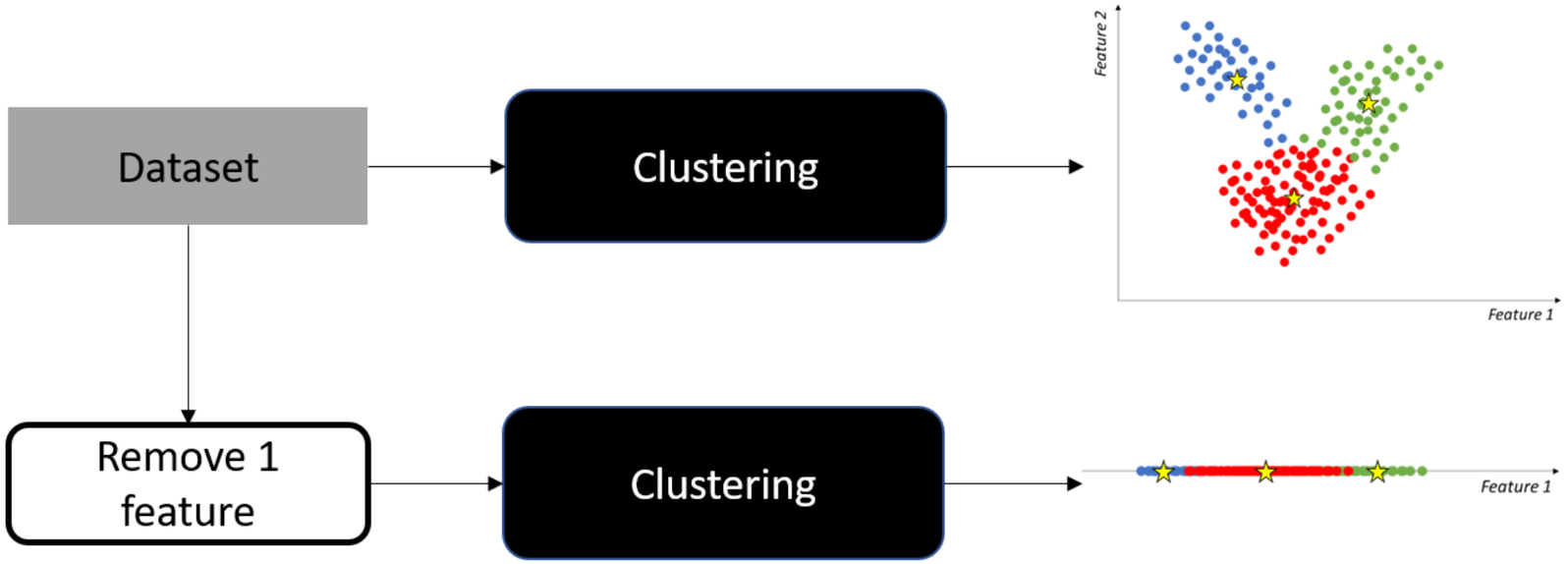
Silhouette score

- Measures clustering's quality
- Ranges from -1 to 1
  - 1 → well-separated clusters
  - -1 → points incorrectly assigned

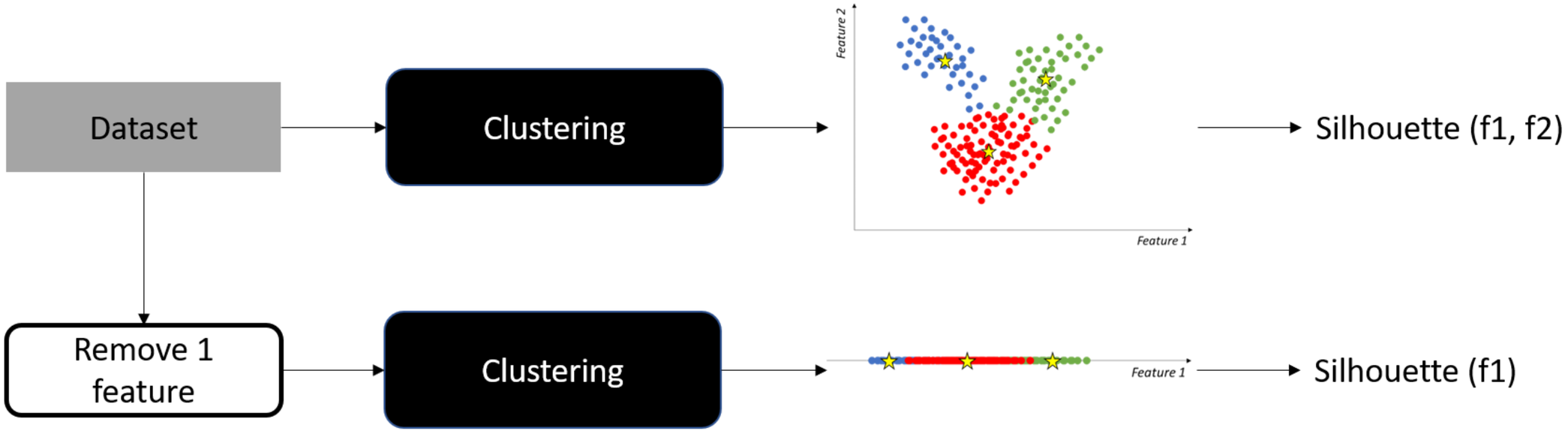


Feature impact on cluster quality

Feature impact on cluster quality



Feature impact on cluster quality



$$\boldsymbol{Impact\ (f2) = Silhouette\ (f1, f2) - Silhouette\ (f1)}$$

- $\mathrm{Impact}(f) > 0$ → positive contribution for f
- $\mathrm{Impact}(f) < 0$ → f introduces noise

Student Performance dataset

age	health status	absences	G1	G2	G3
18	3	4	0	11	11
17	3	2	9	11	11
15	3	6	12	13	12
15	5	0	14	14	14
16	5	0	11	13	13

`X` : array containing features

Computing feature impact on cluster quality

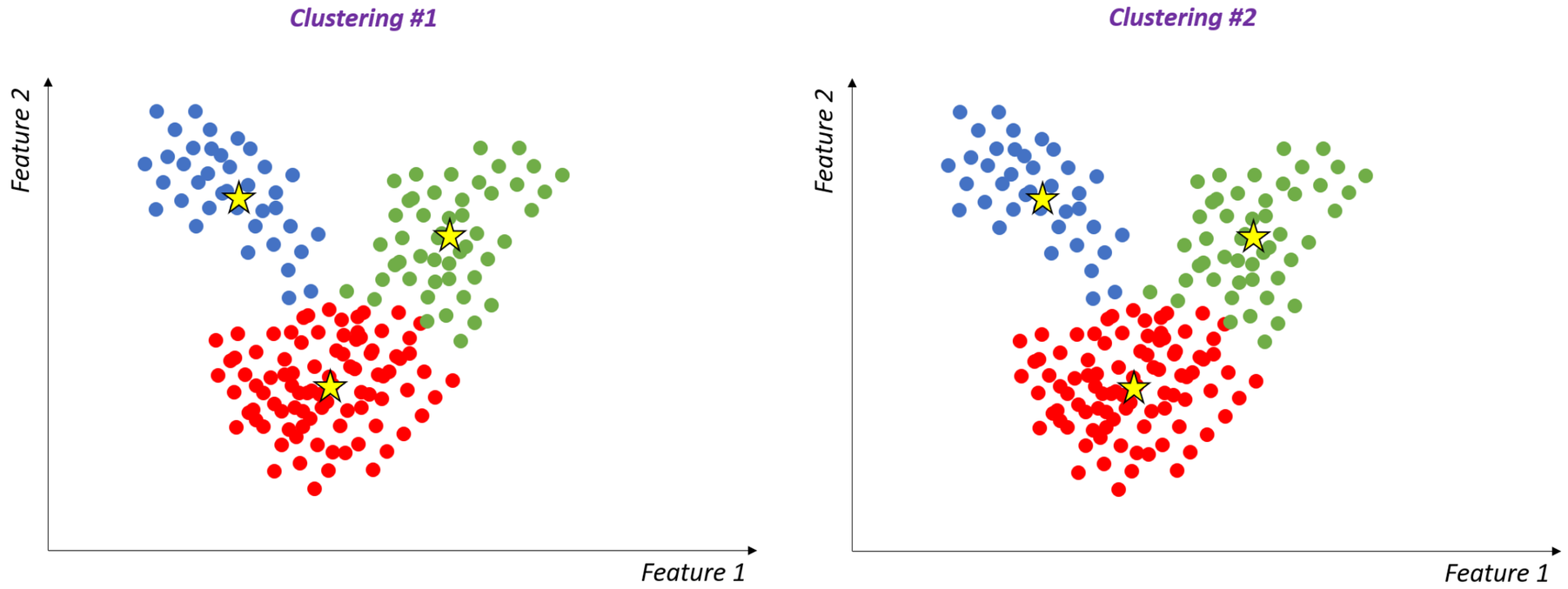
```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

kmeans = KMeans(n_clusters=2).fit(X)
original_score = silhouette_score(X, kmeans.labels_)

for i in range(X.shape[1]):
    X_reduced = np.delete(X, i, axis=1)
    kmeans.fit(X_reduced)
    new_score = silhouette_score(X_reduced, kmeans.labels_)
    impact = original_score - new_score
    print(f'Feature {column_names[i]}: Impact = {impact}')
```

Computing feature impact on cluster quality

```
Feature age: Impact = 0.0519918166274128 1
Feature health status: Impact = 0.0604673742022763 8
Feature absences: Impact = 0.0312909405820266 94
Feature G1: Impact = -0.0257464219406523 53
Feature G2: Impact = -0.0257829233936411 9
Feature G3: Impact = -0.0316341945833015 8
```

Adjusted rand index (ARI)

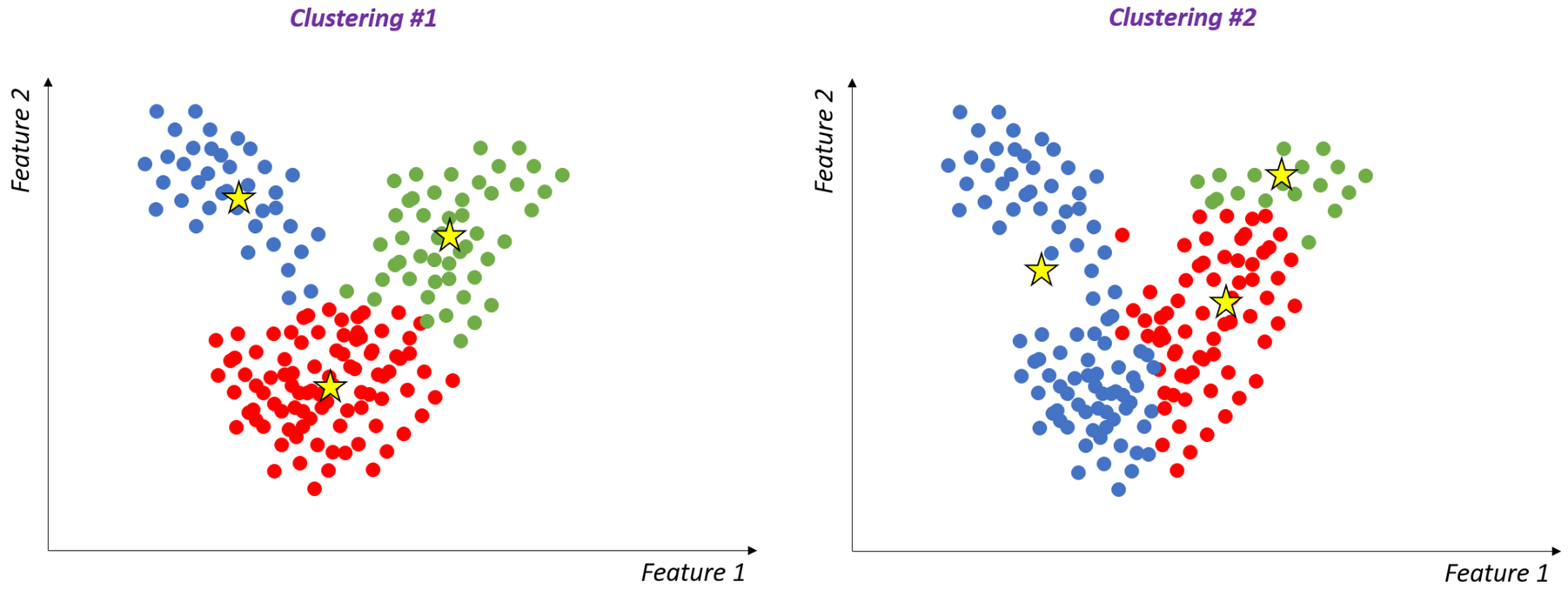
- Measures how well cluster assignments match



- Maximum ARI = 1 → perfect cluster alignment

Adjusted rand index (ARI)

- Measures how well cluster assignments match



- Maximum ARI = 1 → perfect cluster alignment
- Lower ARI → greater difference in clusterings

Feature importance for cluster assignments

- Remove features one at a time
- $\text{Importance}(f) = 1 - \text{ARI}\,(\text{original clusters, modifed clusters})$
- Low (ARI) → high $(1 - ARI)$ → important feature

Feature importance for cluster assignment

```
from sklearn.metrics import adjusted_rand_score

kmeans = KMeans(n_clusters=2).fit(X)
original_clusters = kmeans.predict(X)

for i in range(X.shape[1]):
    X_reduced = np.delete(X, i, axis=1)
    reduced_clusters = kmeans.fit_predict(X_reduced)
    importance = 1 - adjusted_rand_score(original_clusters, reduced_clusters)
    print(f'{df.columns[i]}: {importance}')
```

```
age: 0.0
health status: 0.9995376368119572
absences: 0.0
G1: 0.0
G2: 0.6204069909514572
G3: 0.6204069909514572
```

Let's practice!

EXPLAINABLE AI IN PYTHON

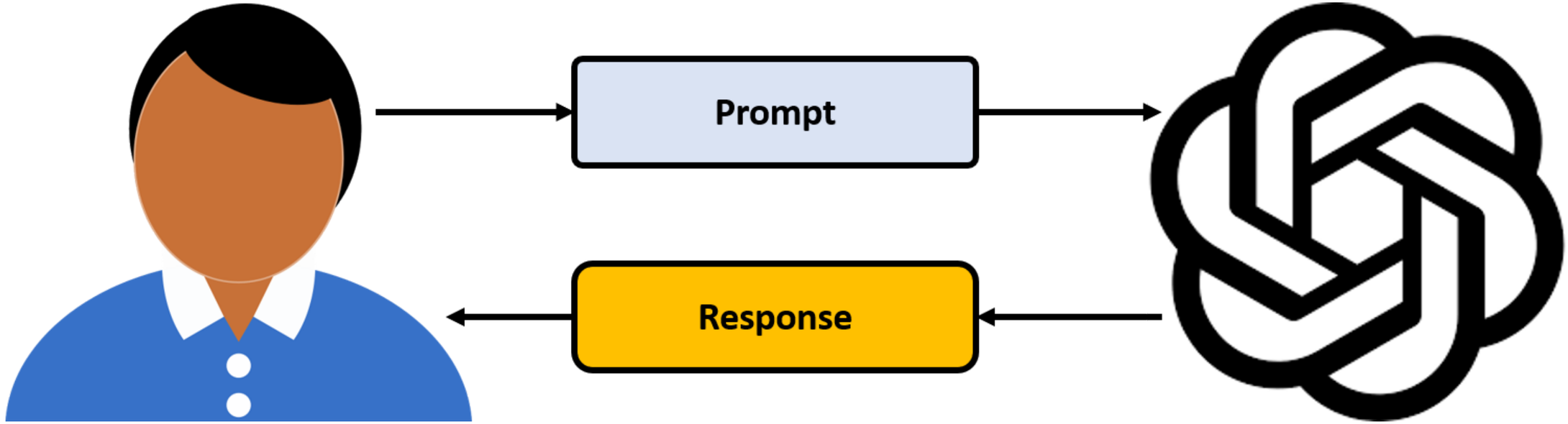
Explaining chat-based generative AI models

EXPLAINABLE AI IN PYTHON

Fouad Trad
Machine Learning Engineer

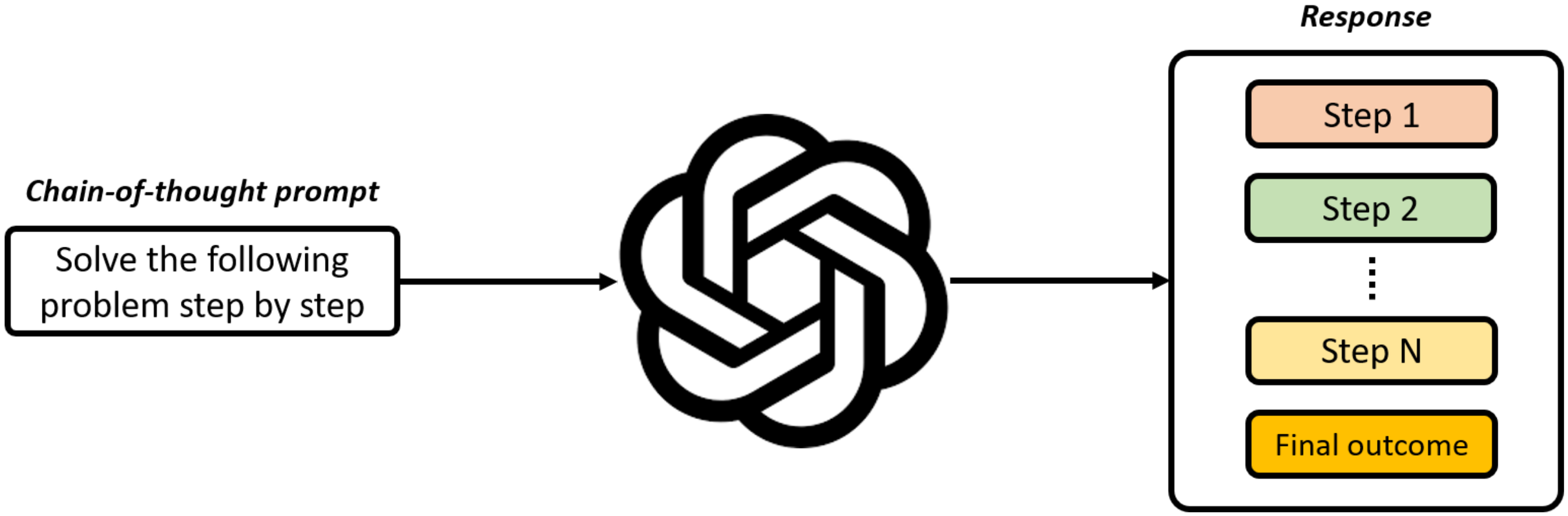
Chat-based generative AI models

- Used to generate text
- Predicts based on context and knowledge



Chain-of-thought prompt

- Encourages model to articulate its reasoning



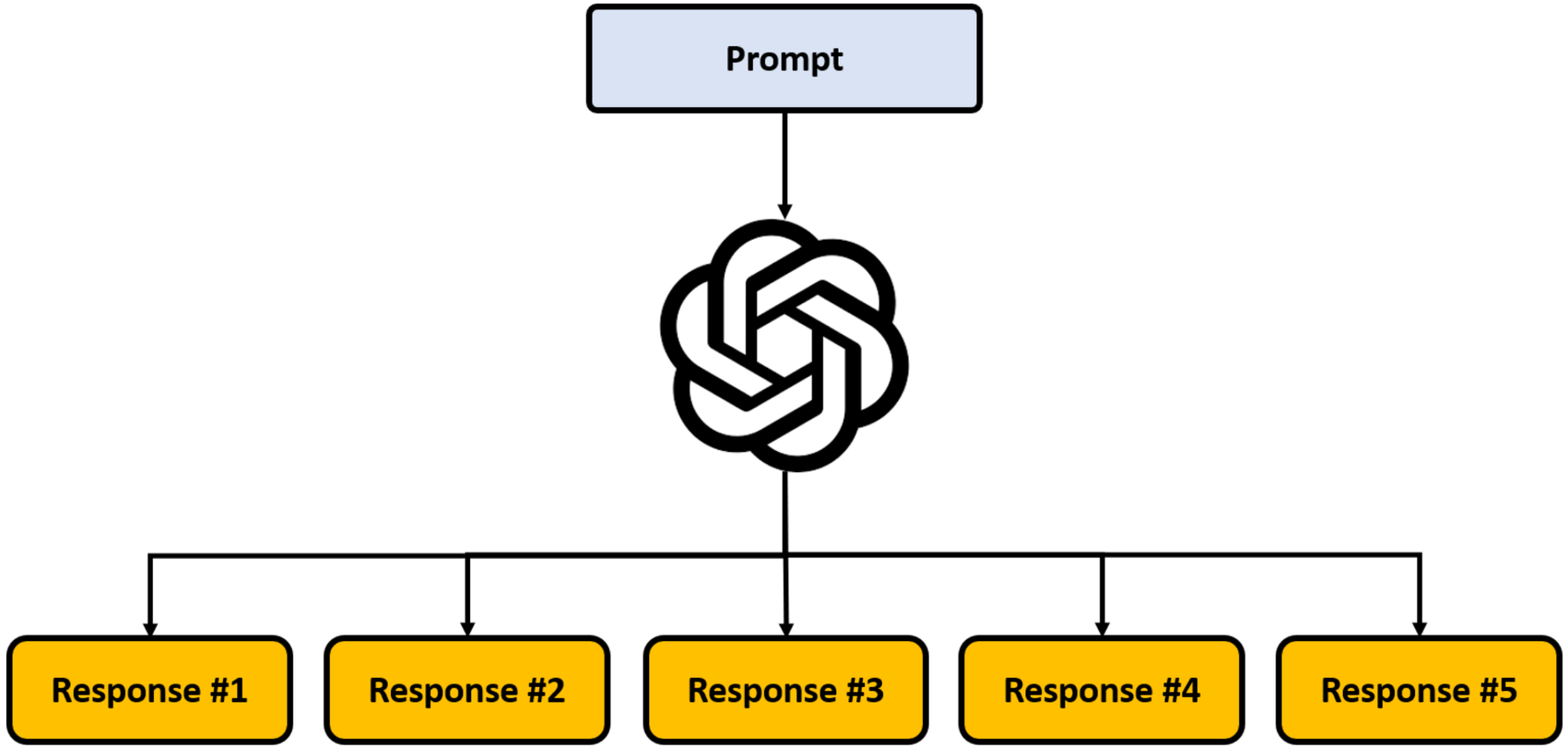
Creating a chain-of-thought prompt

```
prompt = """A shop starts with 20 apples. It sells 5 apples and then receives 8 more.
How many apples does the shop have now? Show your reasoning step-by-step."""
response = get_response(prompt)
print(response)
```

```
To find out how many apples the shop has now, let's follow the transactions step-by-step:
1- The shop begins with 20 apples.
2- The shop sells 5 apples.
   We subtract this number from the starting quantity: 20 - 5 = 15
3- The shop receives 8 apples.
   We add this number to the remaining apples: 15 + 8 = 23
So, after these transactions, the shop has 23 apples.
```

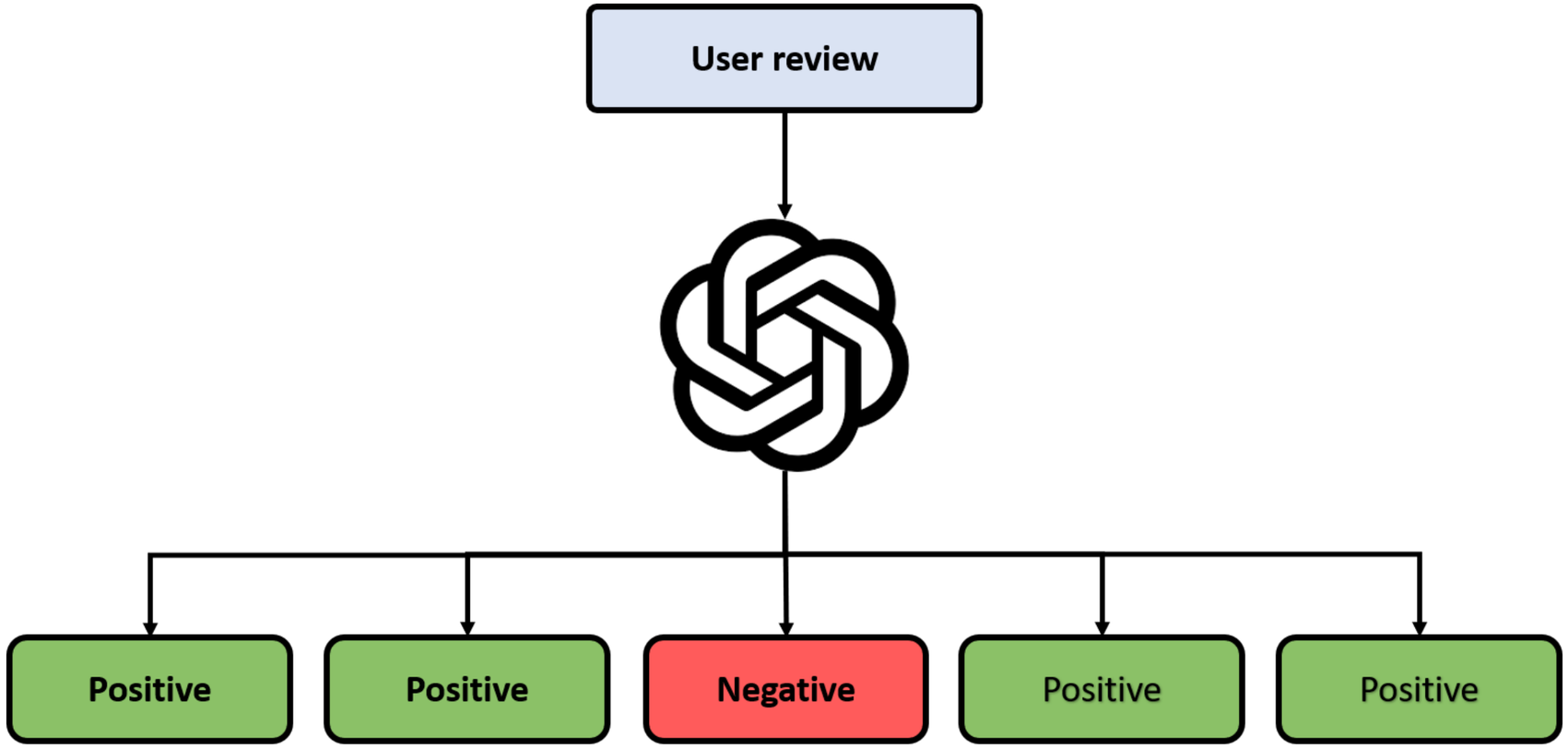
Self-consistency

Assesses model's confidence in generated answers



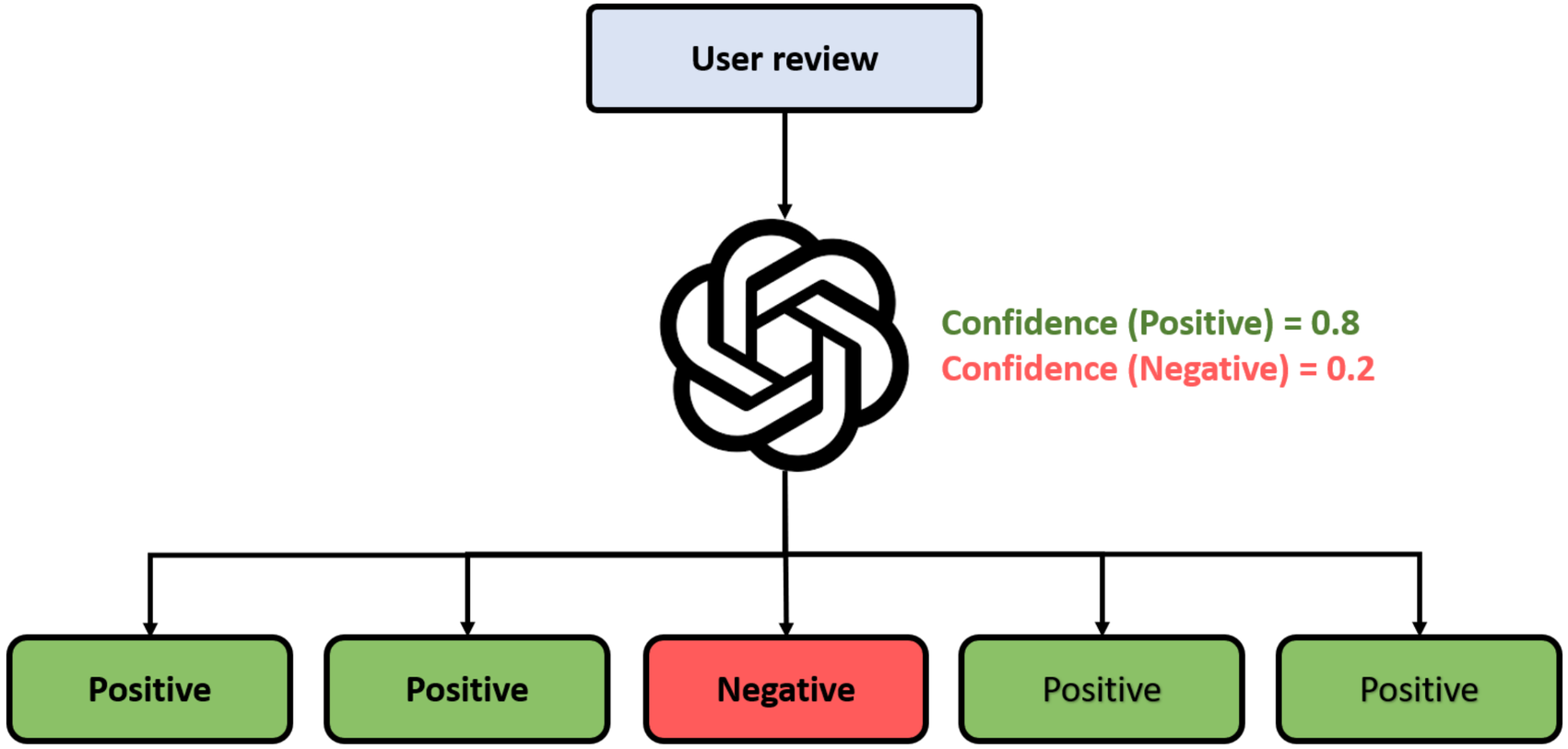
Self-consistency in text classification

Useful for text classification tasks



Self-consistency in text classification

Useful for text classification tasks



Creating self-consistency prompts

```
prompt = """Classify the following review as positive or negative.
You should reply with either "positive" or "negative", nothing else.
Review: 'The customer service was great, but the product itself did not meet my expectations.'"""

responses = []
for i in range(5):
    sentiment = get_response(review)
    responses.append(sentiment.lower())

confidence = {
    'positive': responses.count('positive') / len(responses),
    'negative': responses.count('negative') / len(responses)
}
```

Creating self-consistency prompts

```
print(confidence)
```

```
{
    'positive': 0.6,
    'negative': 0.4
}
```

Let's practice!

EXPLAINABLE AI IN PYTHON

Congratulations

EXPLAINABLE AI IN PYTHON

Fouad Trad
Machine Learning Engineer

Chapter 1

Basics of explainable AI:

- Key principles and terminologies
- Model-specific explainability of linear and tree-based models

Chapter 1

Basics of explainable AI:

- Key principles and terminologies
- Model-specific explainability of linear and tree-based models

Chapter 2

Model-agnostic techniques:

- Permutation importance
- SHAP

Chapter 1

Basics of explainable AI:

- Key principles and terminologies
- Model-specific explainability of linear and tree-based models

Chapter 2

Model-agnostic techniques:

- Permutation importance
- SHAP

Chapter 3

Local explainability:

- SHAP
- LIME

Chapter 1

Basics of explainable AI:

- Key principles and terminologies
- Model-specific explainability of linear and tree-based models

Chapter 2

Model-agnostic techniques:

- Permutation importance
- SHAP

Chapter 3

Local explainability:

- SHAP
- LIME

Chapter 4

Advanced topics:

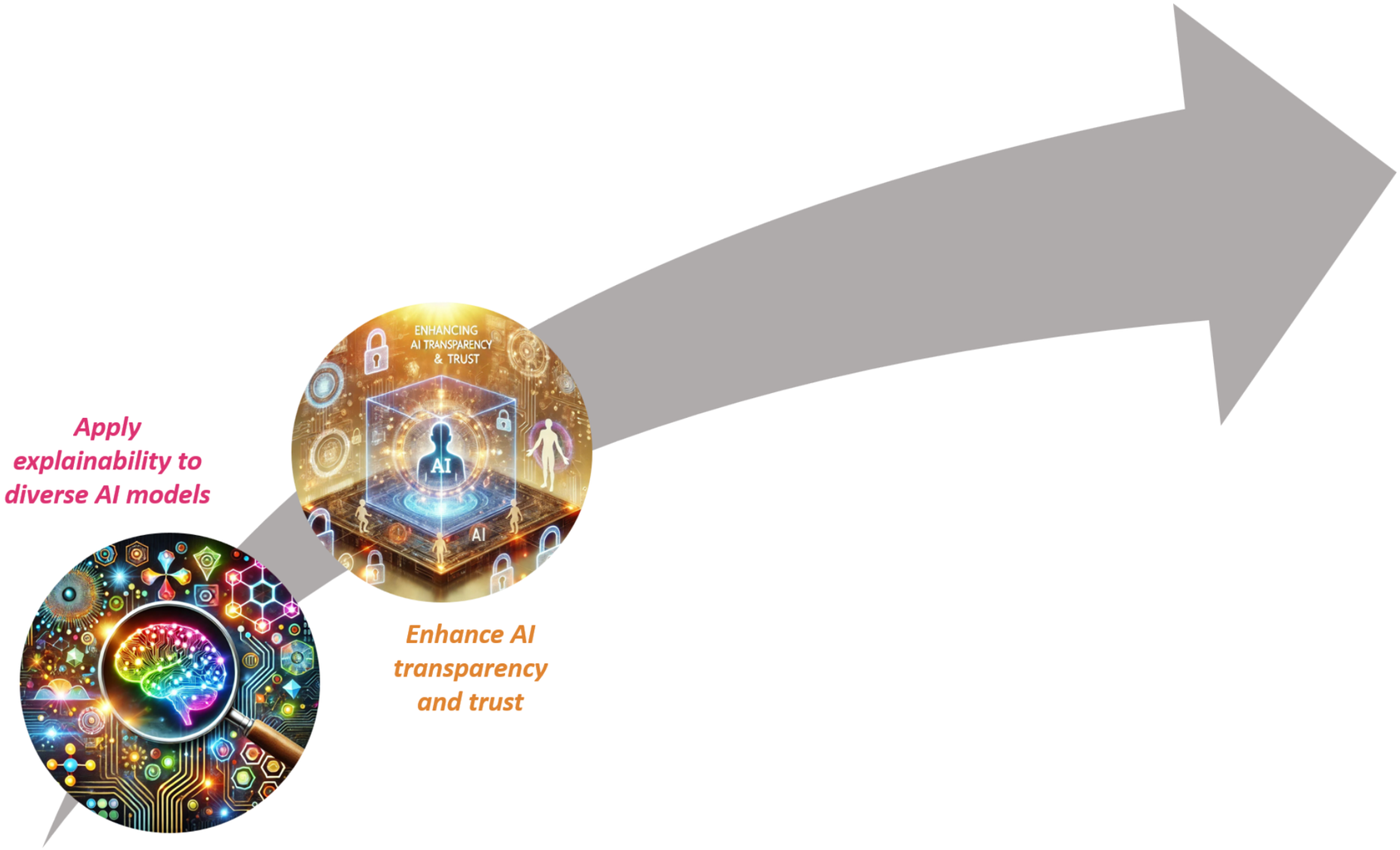
- Explainability metrics
- Explaining unsupervised models
- Explaining generative AI models

What's next?

Apply explainability to diverse AI models

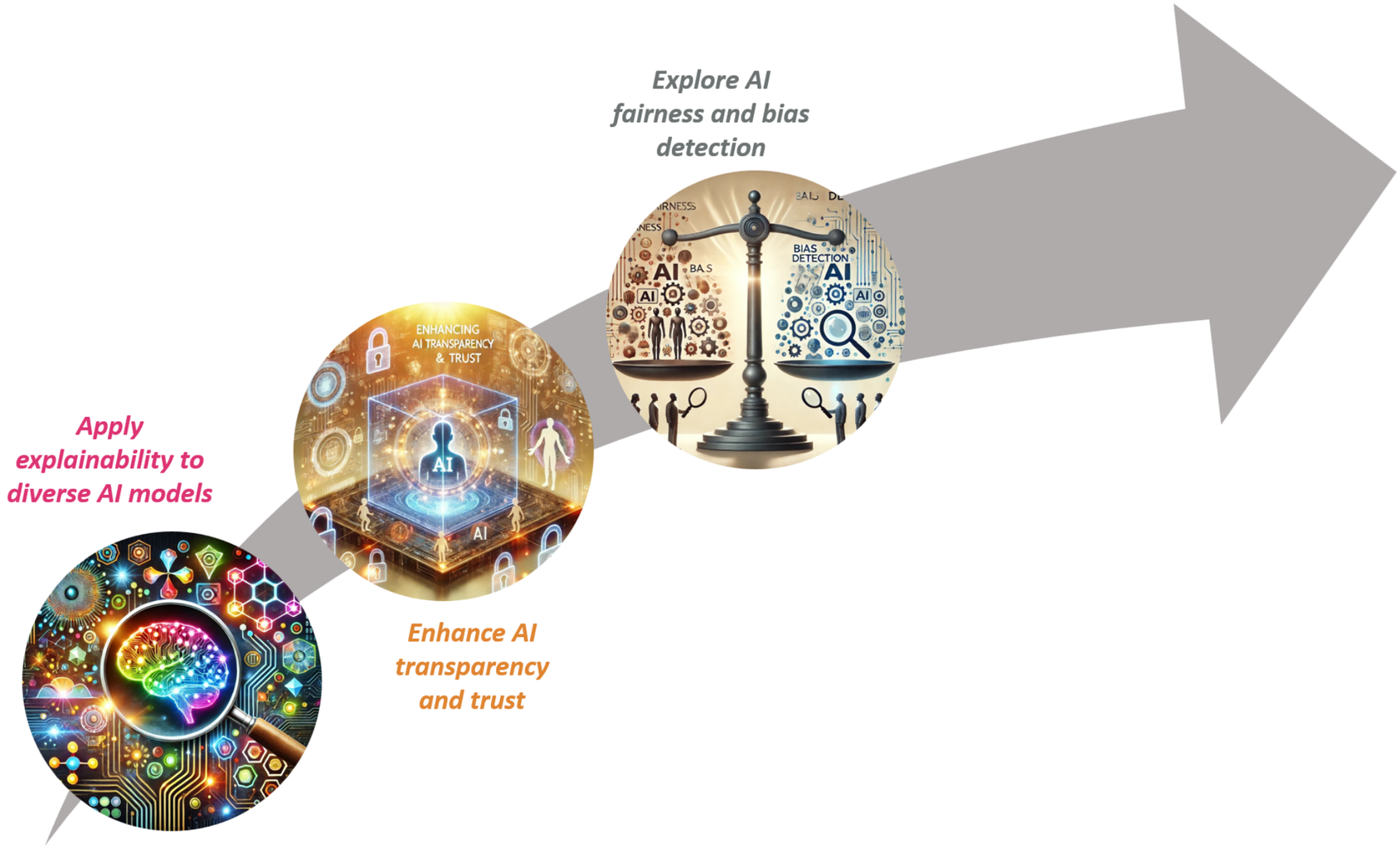
[1] Images generated by DALL-E

What's next?



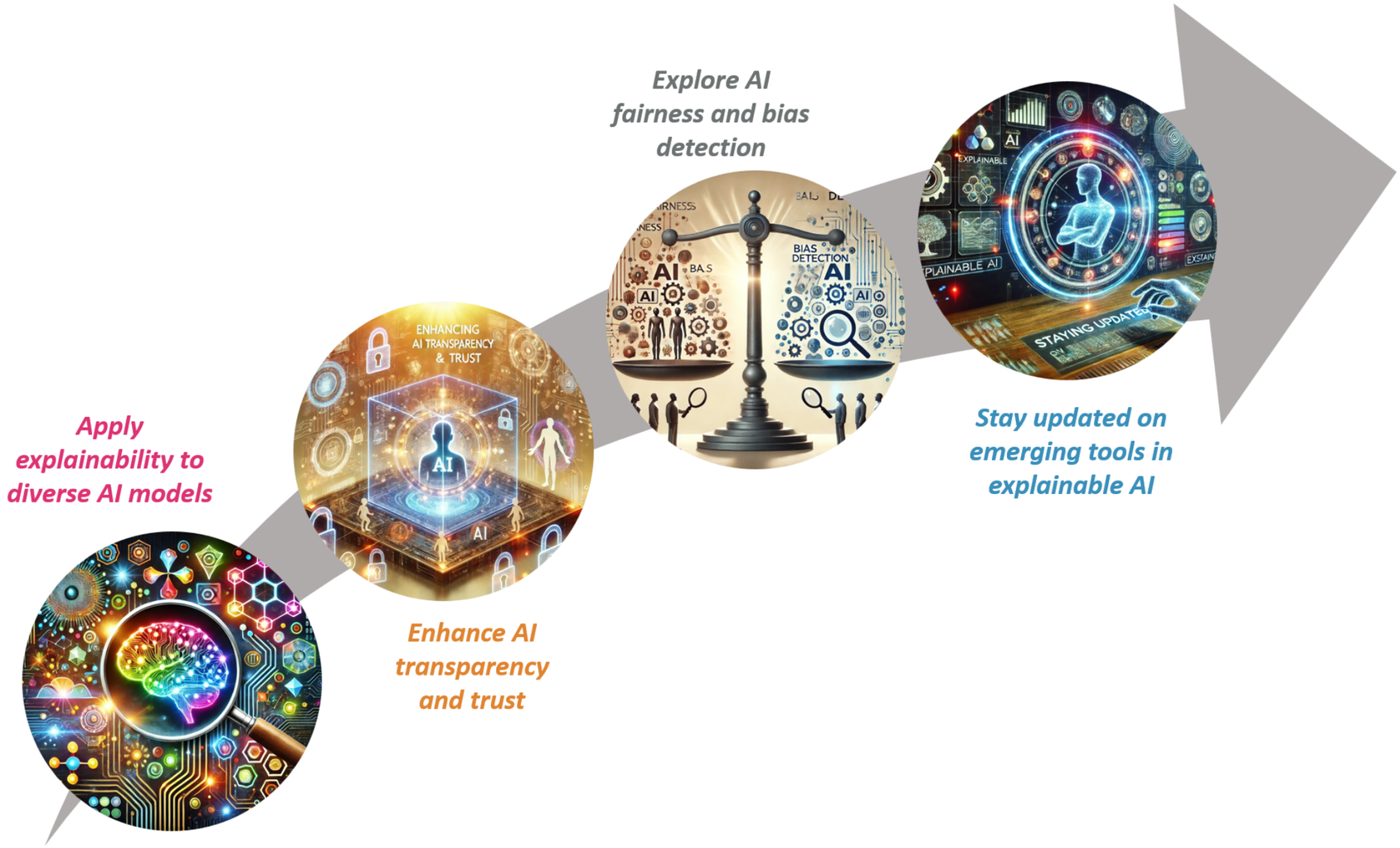
[1] Images generated by DALL-E

What's next?



[1] Images generated by DALL-E

What's next?



[1] Images generated by DALL-E

Congratulations!

EXPLAINABLE AI IN PYTHON