Introduction to explainable AI

EXPLAINABLE AI IN PYTHON

Fouad Trad
Machine Learning Engineer

datacamp

About me

Fouad Trad

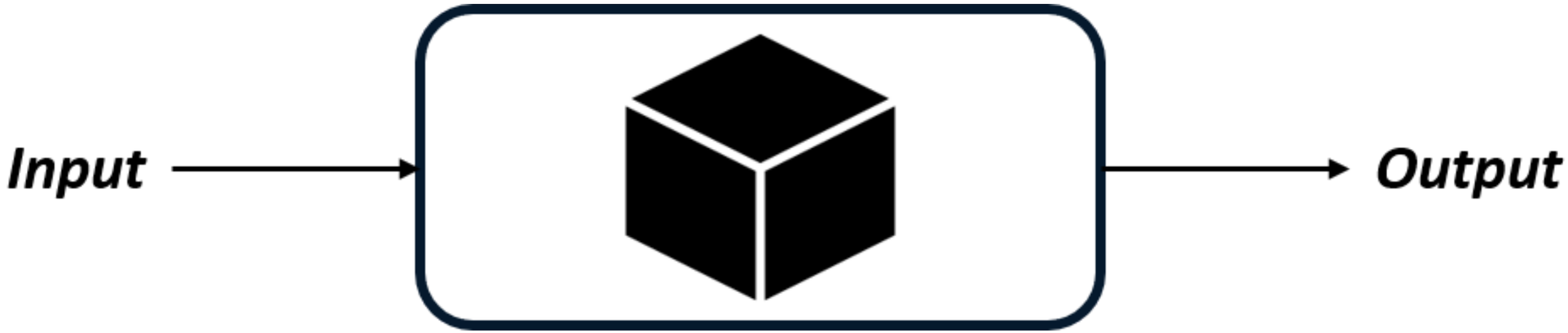
- Machine learning engineer
- PhD Candidate
- Specializing in AI for Cybersecurity

LinkedIn

Artificial intelligence

- AI enables machines to perform human-like tasks

The need for explainability



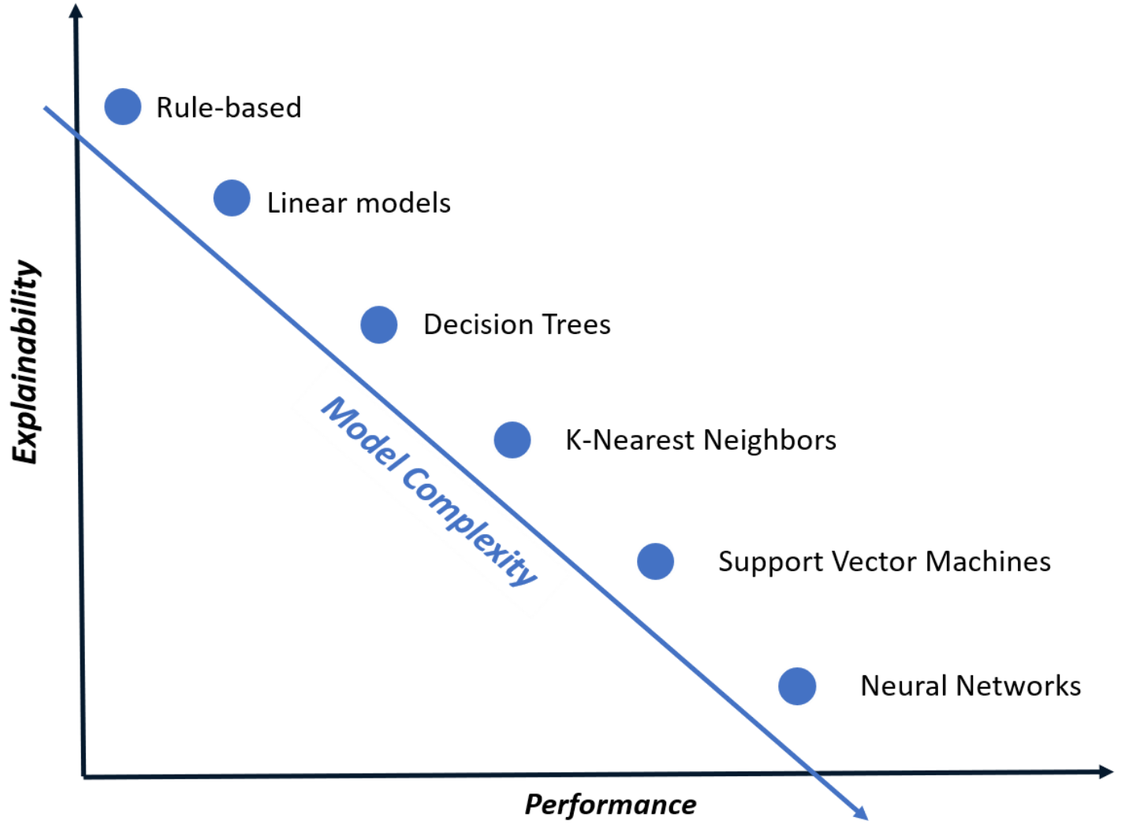
- **Black box AI:** no understanding of decision-making

The need for explainability

- **Black box AI:** no understanding of decision-making
- **Explainable AI:** understand model's inner workings

Explainability vs. accuracy

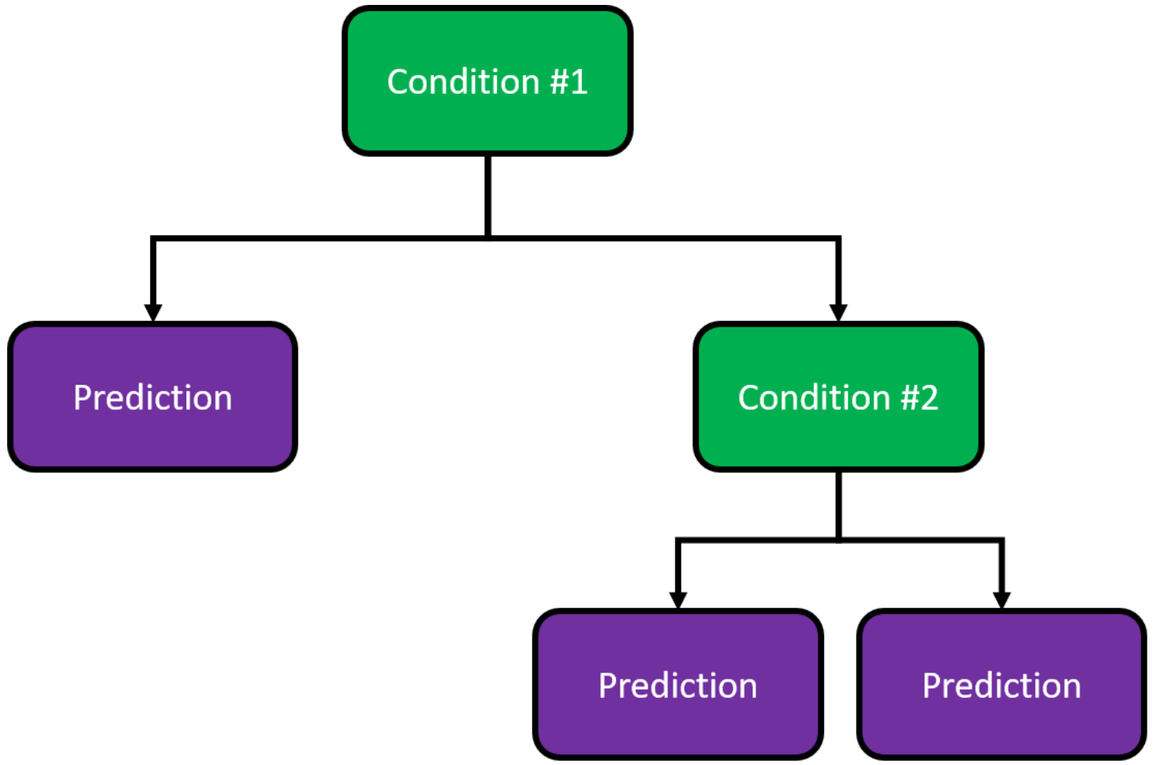
- Basic models are:
  - More explainable
  - Less precise
- Complex models are:
  - More precise
  - Less explainable



Decision trees vs. neural networks

Decision trees

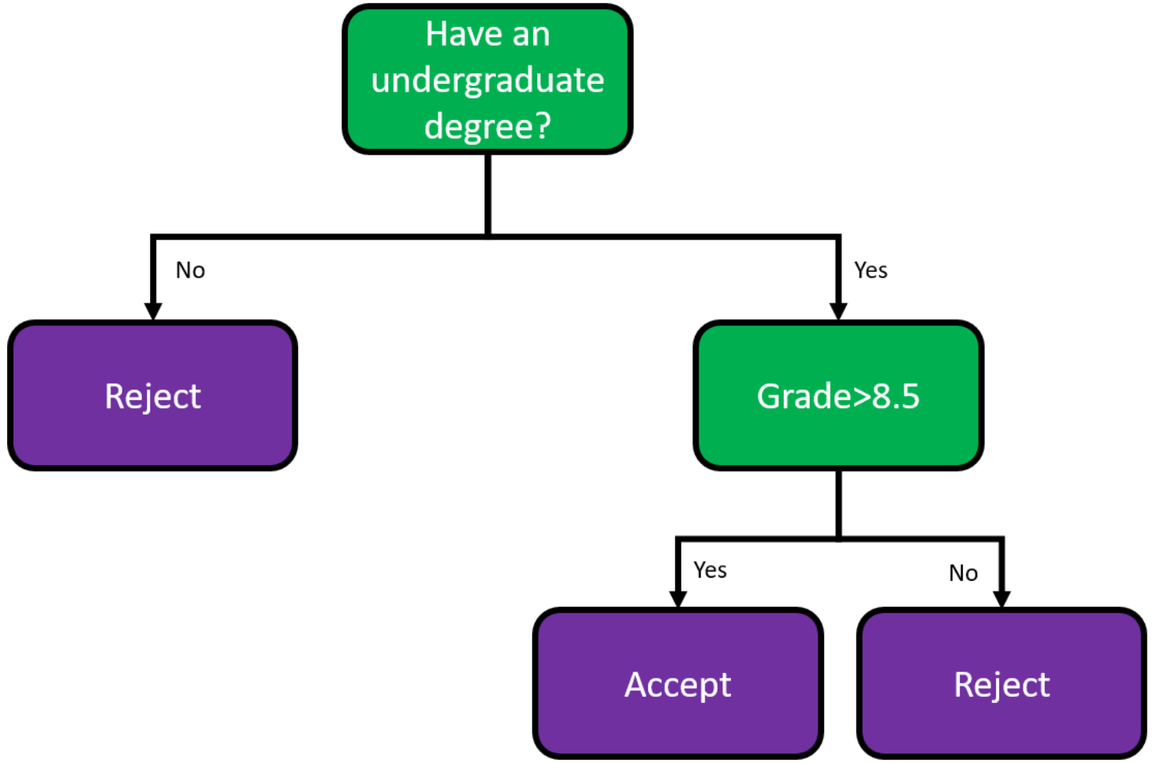
Show decision path based on conditions



Decision trees vs. neural networks

Decision trees

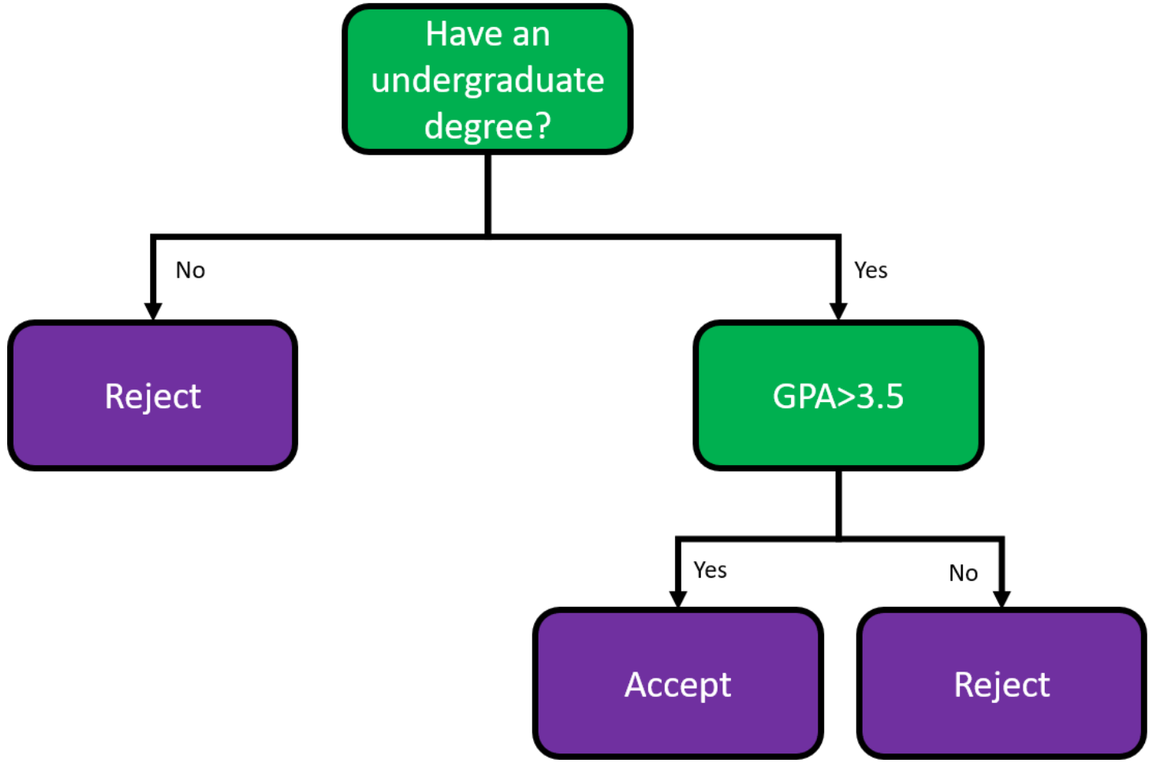
Show decision path based on conditions



Decision trees vs. neural networks

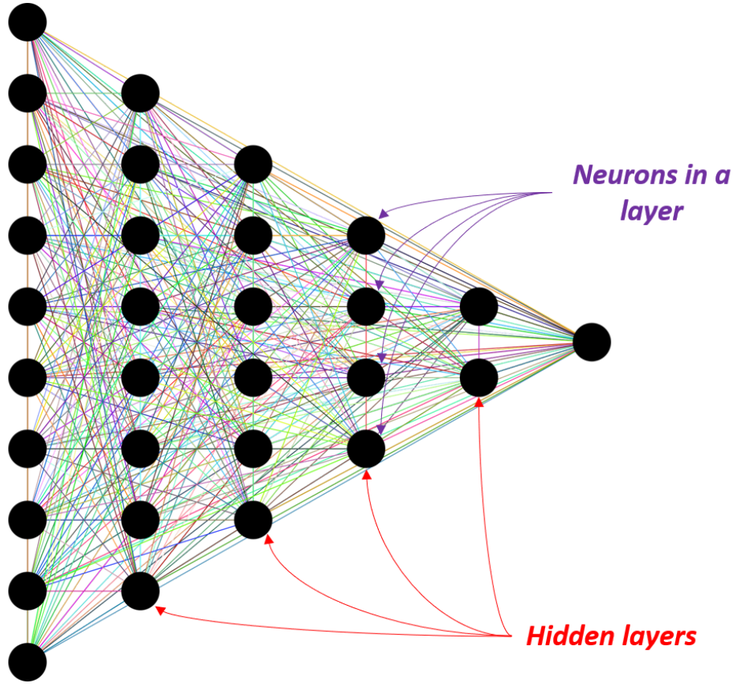
Decision trees

Show decision path based on conditions



Neural networks

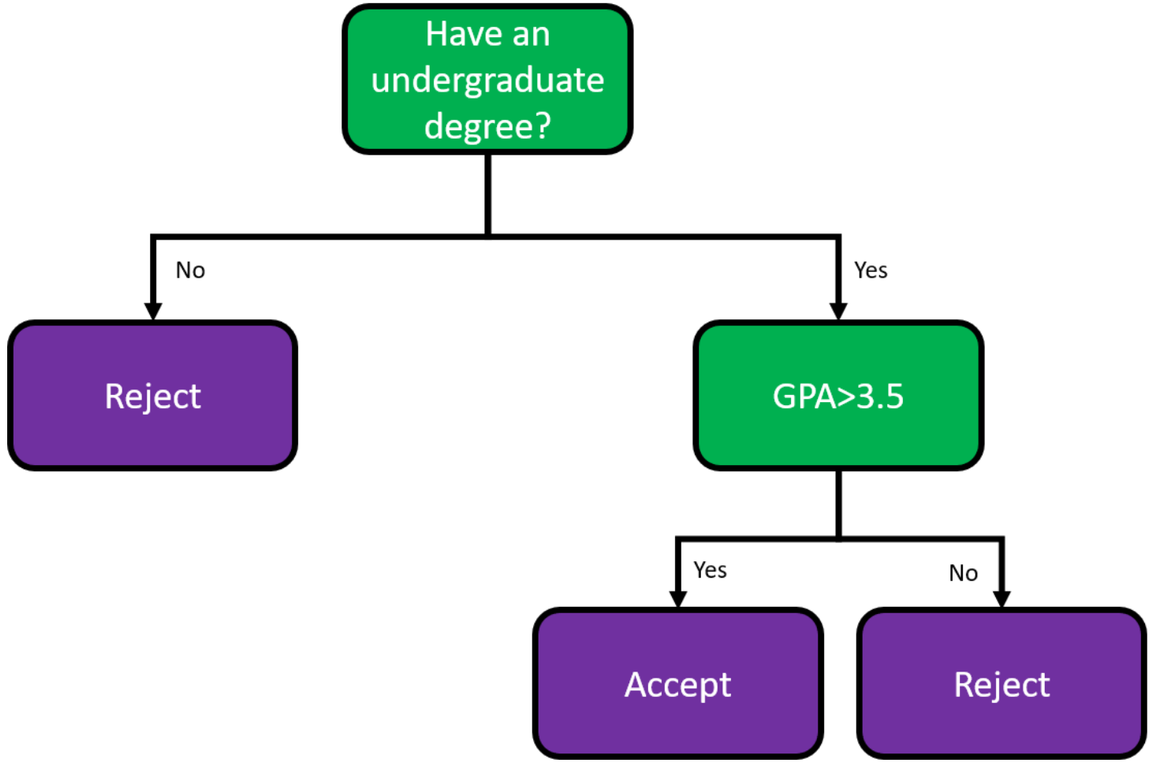
Not inherently transparent



Decision trees vs. neural networks

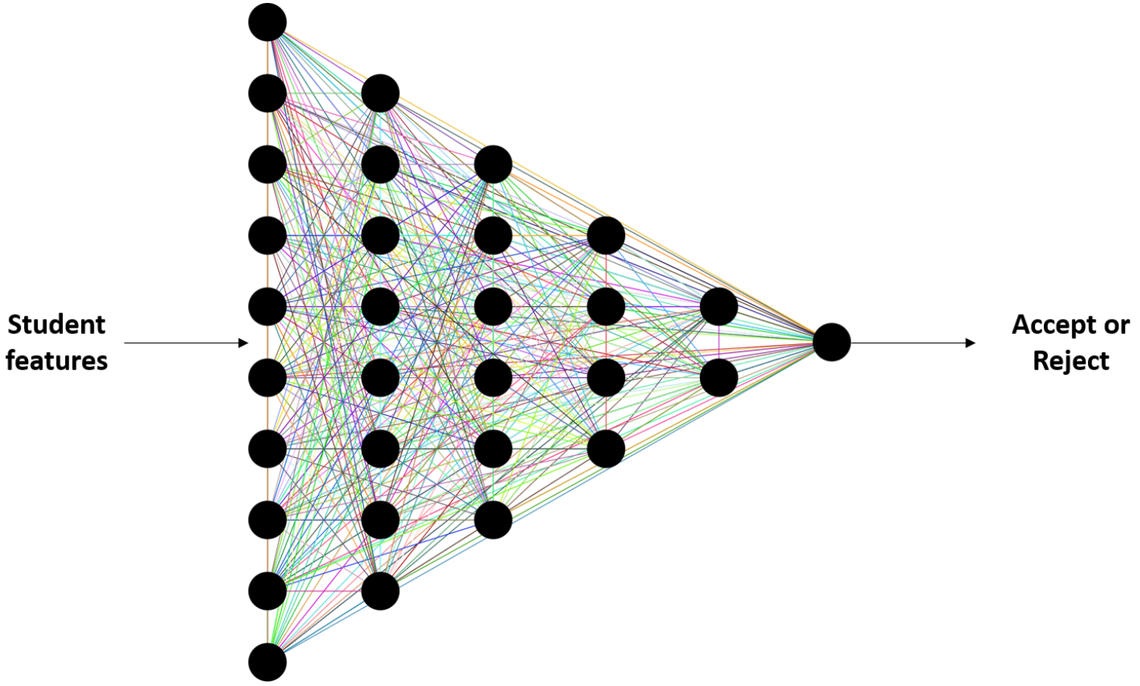
Decision trees

Show decision path based on conditions



Neural networks

Not inherently transparent



Student admission prediction

GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Accept
337	118	4	4.5	4.5	9.65	1
316	104	3	3	3.5	8.00	1
314	103	2	2	3	8.21	0

- Test scores: GRE and TOEFL
- University rating
- SOP: statement of purpose
- LOR: letter of recommendation
- CGPA: cumulative grade point average

Student admission prediction

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(max_depth=5)

model.fit(X_train, y_train)

y_pred = model.predict(y_test)

acc = accuracy_score(y_test, y_pred)
print(f"Accuracy of Decision Tree: {acc}")
```

```
Accuracy of Decision Tree: 0.82
```

Neural Network

```
from sklearn.neural_network import MLPClassifier

model = MLPClassifier(hidden_layer_sizes=(1000,1000))

model.fit(X_train, y_train)

y_pred = model.predict(y_test)

acc = accuracy_score(y_test, y_pred)
print(f"Accuracy of Neural Network: {acc}")
```

```
Accuracy of Neural Network: 0.9
```

Decision tree rules

```
from sklearn.tree export_text

rules = export_text(model, feature_names=list(X_train.columns))

print(rules)
```

```
|--- CGPA <= 8.34
|   |--- GRE Score <= 320.50
|   |   |--- class: 0
|   |--- GRE Score >  320.50
|   |   |--- class: 1
|--- CGPA >  8.34
|   |--- GRE Score <= 319.50
|   |   |--- class: 1
...
```

Model-specific vs. model-agnostic techniques

Model-specific

Apply to particular model

Model-specific vs. model-agnostic techniques

Model-specific

Apply to particular model

Model-agnostic

Apply to any model

What's next?

- Model-specific and model-agnostic techniques
- Local and global methods
- Advanced topics:
  - Explainability metrics
  - Unsupervised explainability
  - Generative AI explainability

Let's practice!

EXPLAINABLE AI IN PYTHON

Explainability in linear models

EXPLAINABLE AI IN PYTHON

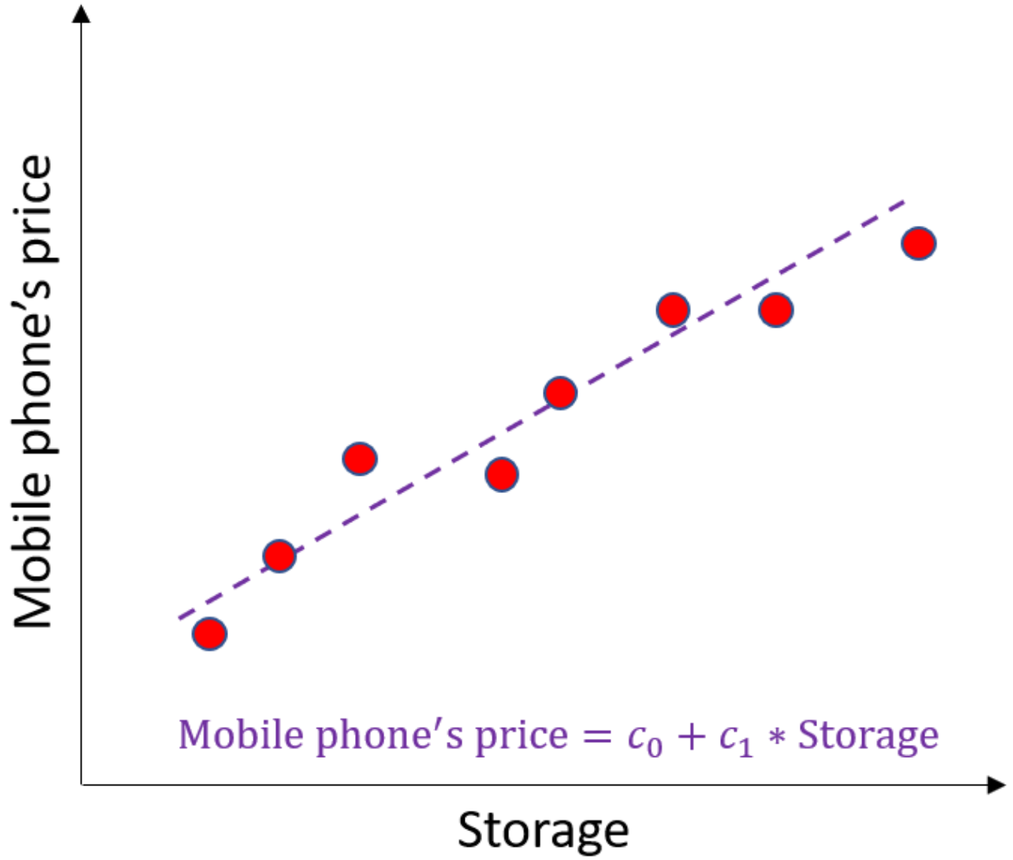
Fouad Trad
Machine Learning Engineer

datacamp

Linear models

Linear regression

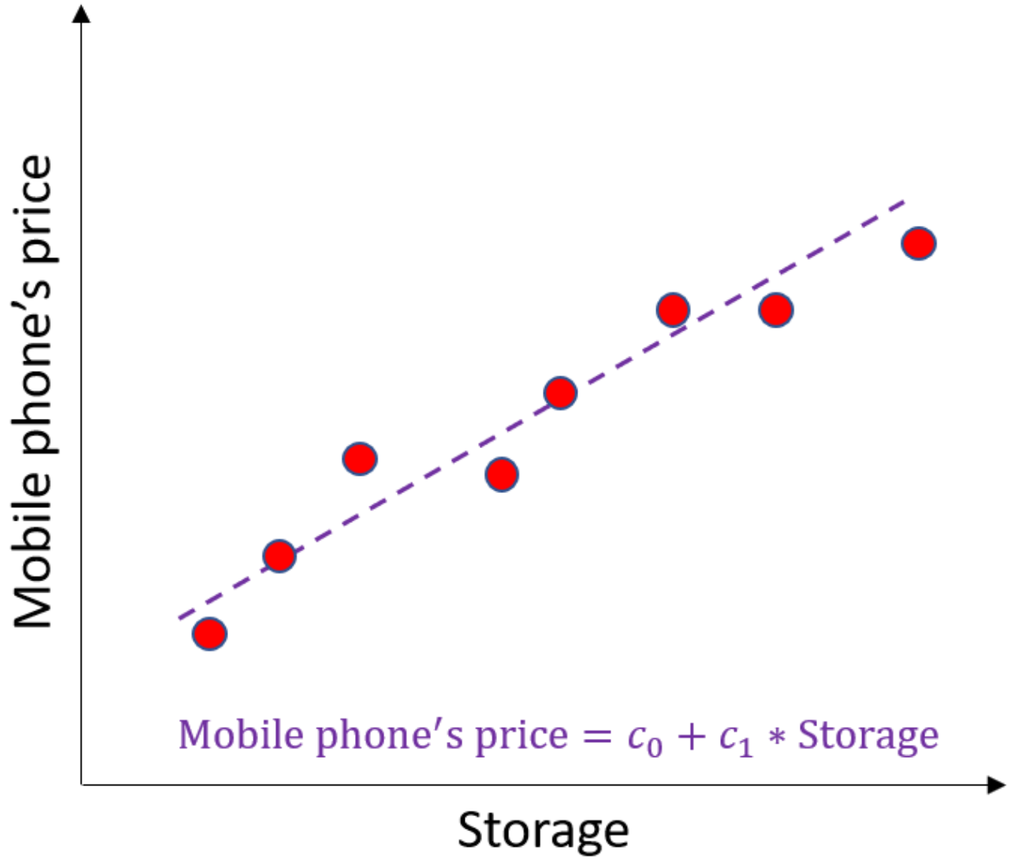
- Predicts continuous values



Linear models

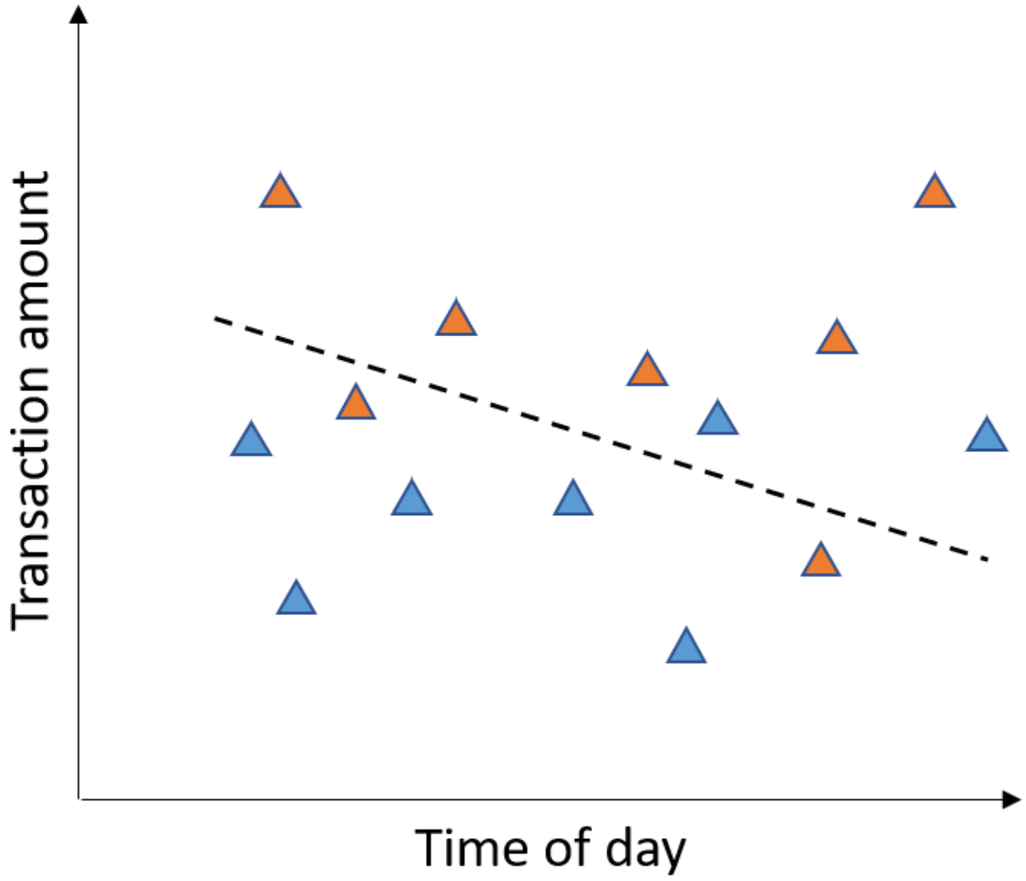
Linear regression

- Predicts continuous values



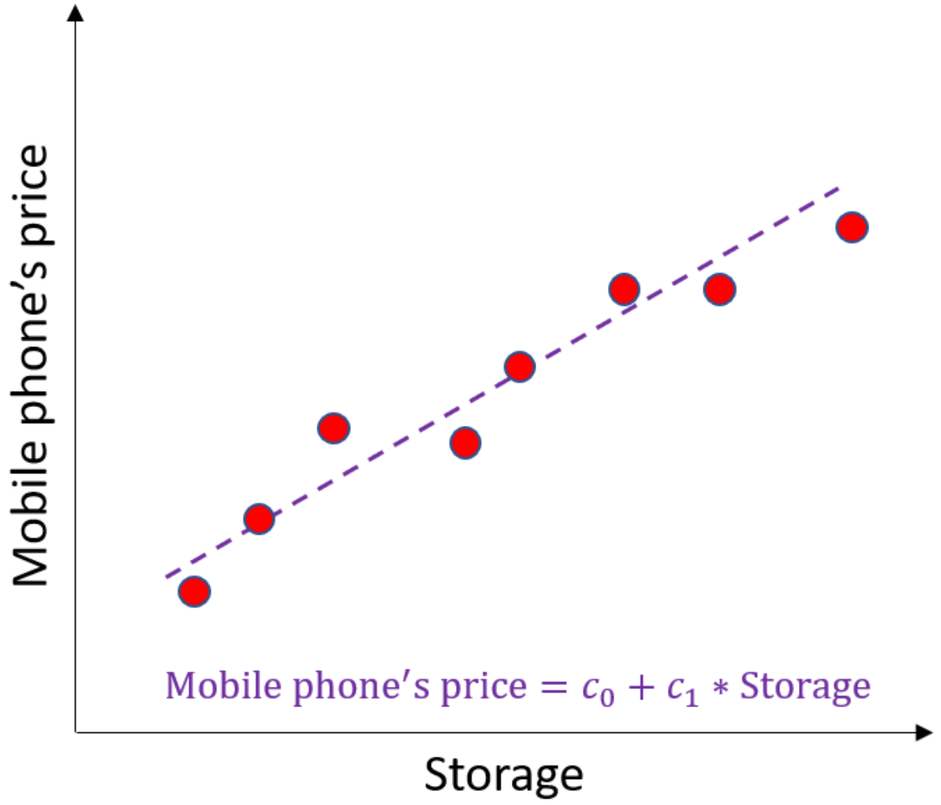
Logistic regression

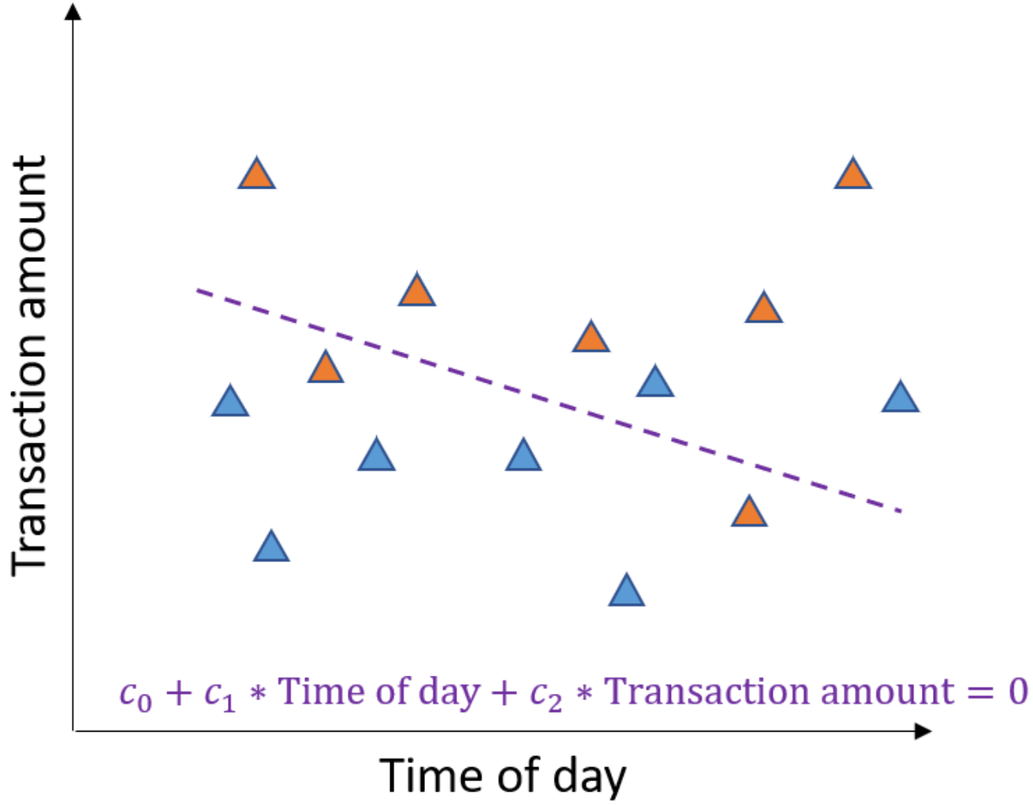
- Used for binary classification



Why are linear models explainable?

- Learn linear combination of input features
- $c_0 + c_1 \times \text{feature}_1 + c_2 \times \text{feature}_2 + \ldots + c_n \times \text{feature}_n$





Coefficients

- Tell us importance of each feature
  - Higher absolute value → higher importance
  - Lower absolute value → lower importance
- To compare coefficients → absolute values
- **Note:** Normalize feature scales before computing coefficients

$$\text{Mobile Phone's price} = c_0 + 5 * Storage + 2 * \text{Number of camera lenses}$$

Coefficients

- Tell us importance of each feature
  - Higher absolute value → higher importance
  - Lower absolute value → lower importance
- To compare coefficients → absolute values
- **Note:** Normalize feature scales before computing coefficients

$$\text{Mobile Phone's price} = c_0 + c_1 * \underbrace{Storage}_{*\,[100\ -1000]GB} + c_2 * \underbrace{\text{Number of camera lenses}}_{<5}$$

Admissions

GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Chance of Admit	Accept
337	118	4	4.5	4.5	9.65	0.92	1
324	107	4	4	4.5	8.87	0.76	1
316	104	3	3	3.5	8	0.72	1
322	110	3	3.5	2.5	8.67	0.8	1
314	103	2	2	3	8.21	0.45	0

```
X_train = data.drop(['Chance of Admit', 'Accept'], axis=1)
y_reg = data['Chance of Admit']
y_cls = data['Accept']
```

Model training

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LinearRegression, LogisticRegression

scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(X_train)

lin_reg = LinearRegression()
lin_reg.fit(X_train_scaled, y_reg)

log_reg = LogisticRegression()
log_reg.fit(X_train_scaled, y_cls)
```

Computing coefficients

Linear regression

```
print(lin_reg.coef_)
```

```
[0.03052087 0.01665433 0.00668971
 0.00326926 0.01724815 0.0661691 ]
```

Logistic regression

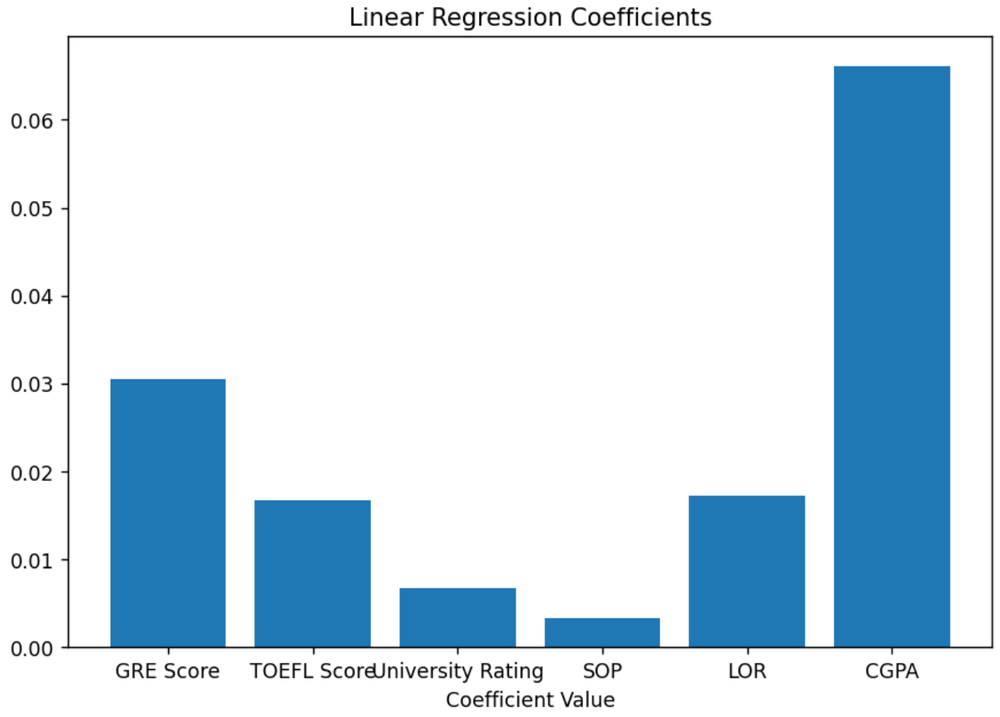
```
print(log_reg.coef_)
```

```
[[1.28985577  0.49441086  0.47593379
  0.05434322  0.41800927  1.31980189]]
```

Visualizing coefficients

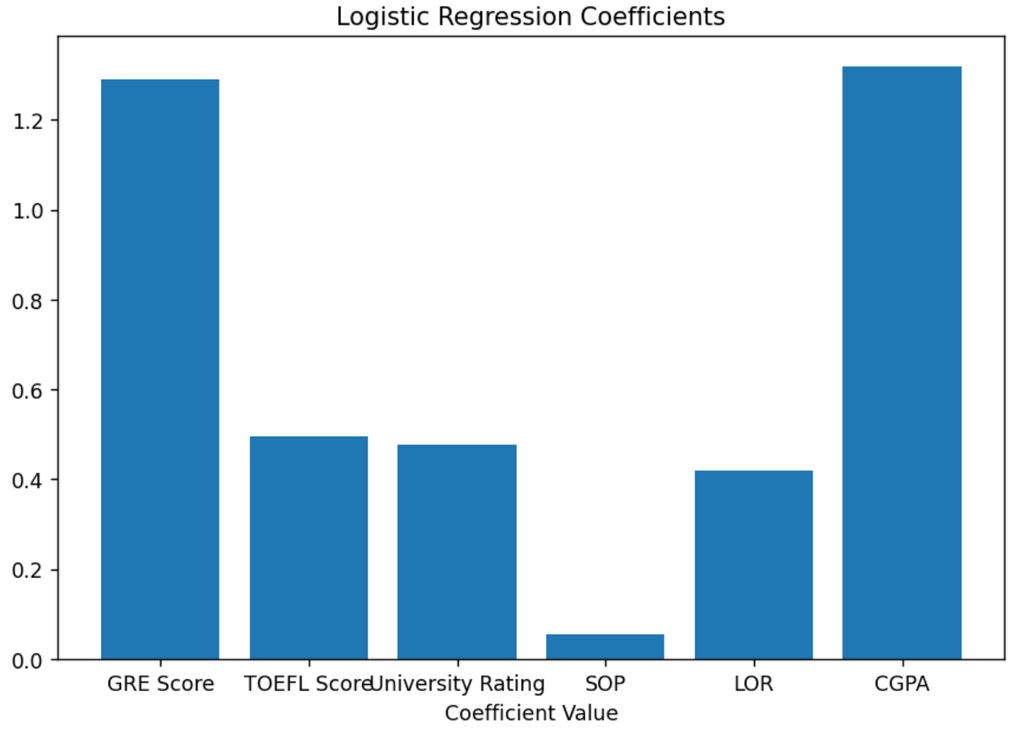
Linear regression

```
import matplotlib.pyplot as plt
plt.bar(X_train.columns, lin_reg.coef_)
```



Logistic regression

```
import matplotlib.pyplot as plt
plt.bar(X_train.columns, log_reg.coef_[0])
```



Let's practice!

EXPLAINABLE AI IN PYTHON

Explainability in tree-based models

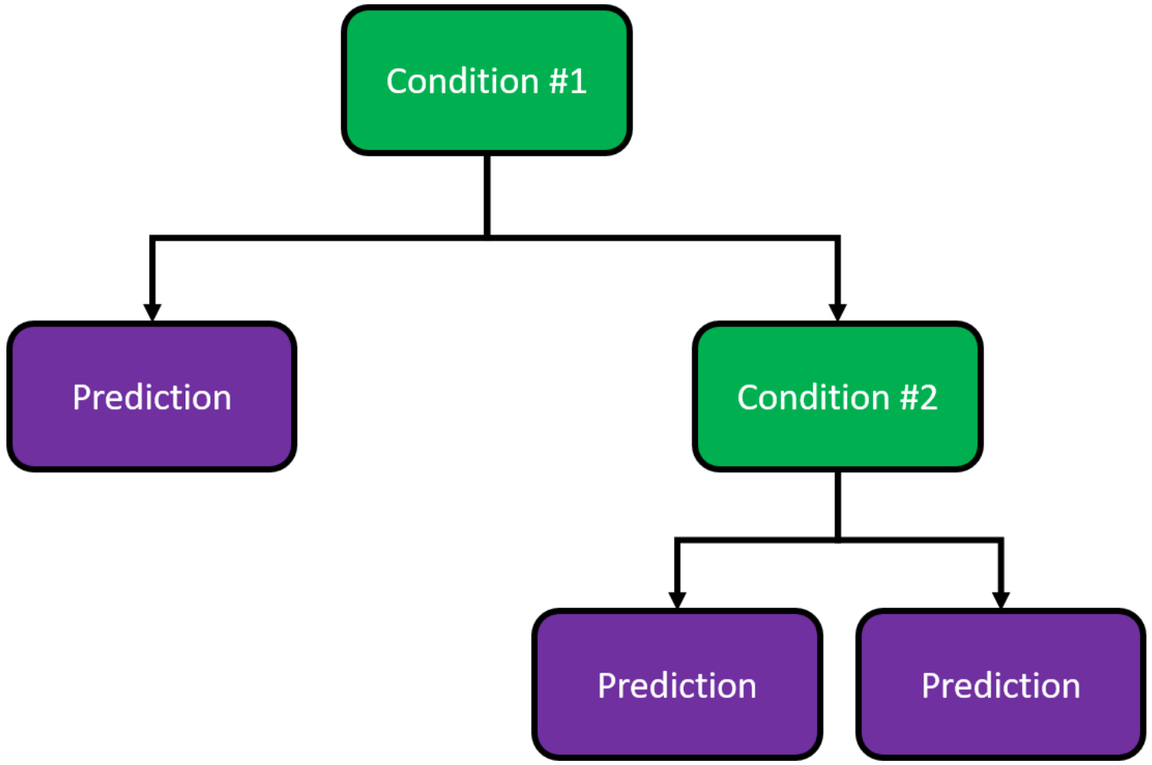
EXPLAINABLE AI IN PYTHON

Fouad Trad
Machine Learning Engineer

datacamp

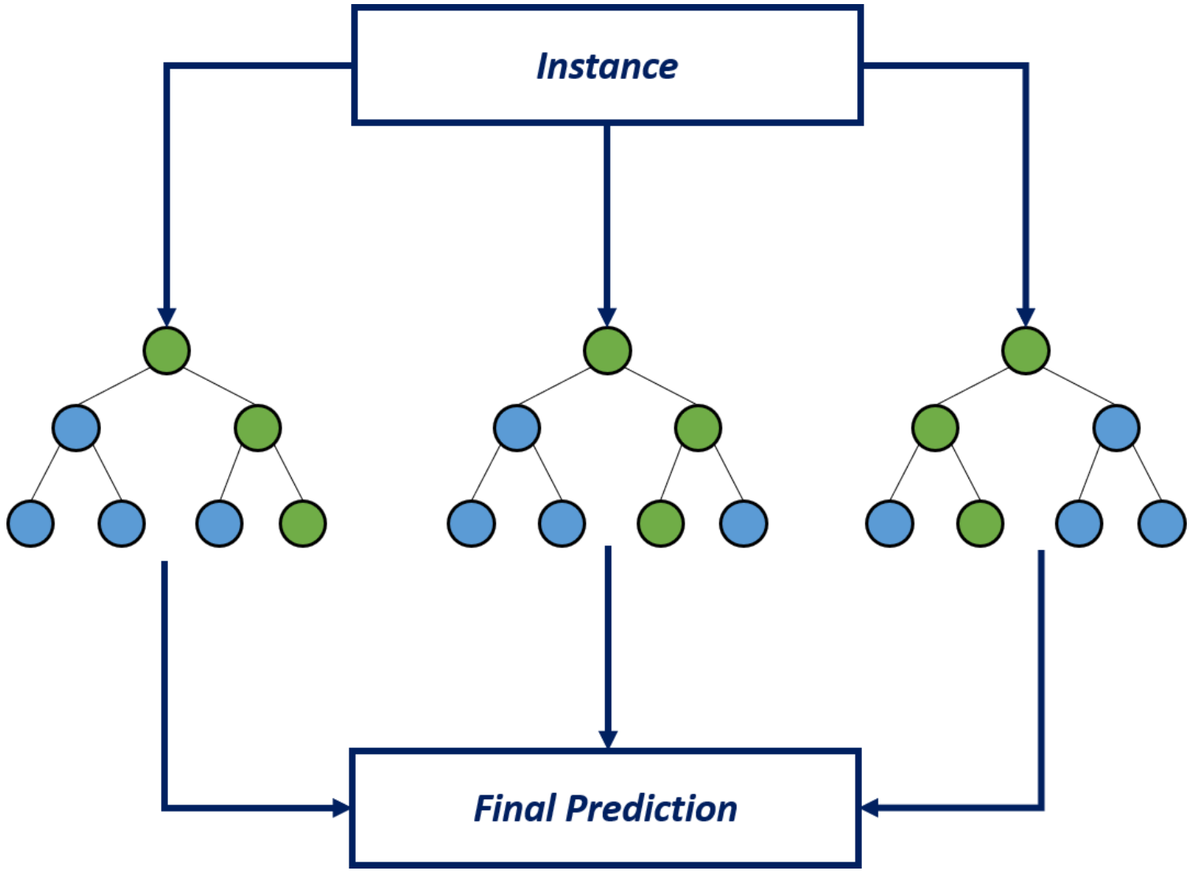
Decision tree

- Fundamental block of tree-based models
- Used for regression and classification
- Tree-like structure for predictions
  - Several decisions
  - Each decision is based on one feature
- Inherently explainable



Random forest

- Consists of many decision trees
- Used for regression and classification
- Complicates direct interpretability
- Feature importance
  - Measures reduction of uncertainty in predictions
  - Different than coefficients in linear models



Admissions dataset

GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Accept
337	118	4	4.5	4.5	9.65	1
324	107	4	4	4.5	8.87	1
316	104	3	3	3.5	8.00	1
322	110	3	3.5	2.5	8.67	1
314	103	2	2	3	8.21	0

```
X_train = data.drop(['Accept'], axis=1)
y_train = data['Accept']
```

Model training

```
from sklearn.tree import DecisionTreeClassifier

tree_model = DecisionTreeClassifier()
tree_model.fit(X_train, y_train)

print(tree_model.feature_importances_)
```

```
[0.17936982 0.08878744 0.04388924
 0.0532897  0.07130751 0.56335628]
```

```
from sklearn.ensemble import RandomForestClassifier

forest_model = RandomForestClassifier()
forest_model.fit(X_train, y_train)

print(forest_model.feature_importances_)
```

```
[0.25347149 0.17518662 0.06551317
 0.06758647 0.07866478 0.35957747]
```

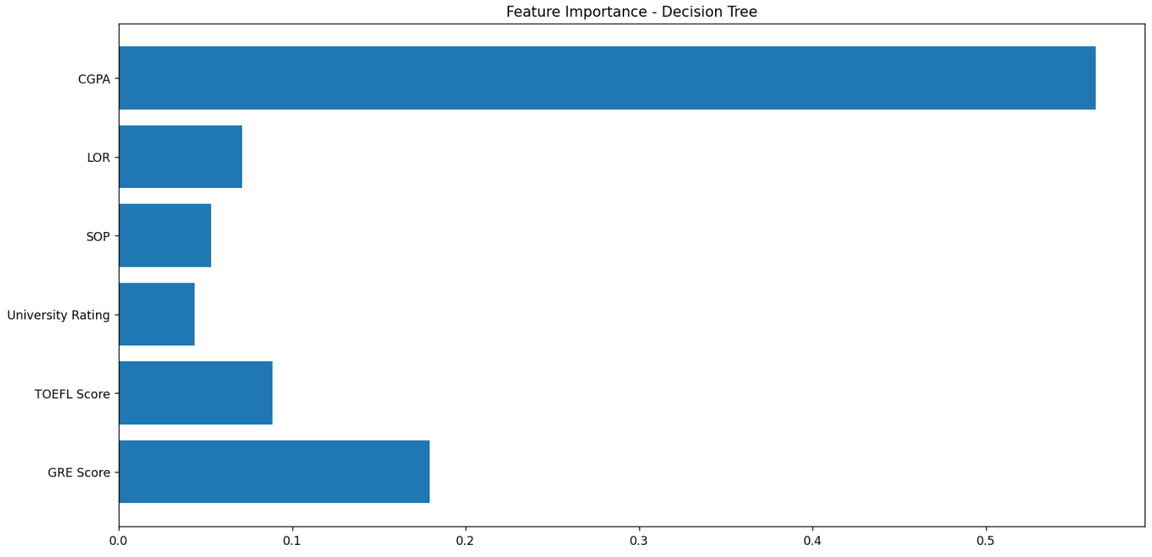
Feature importance

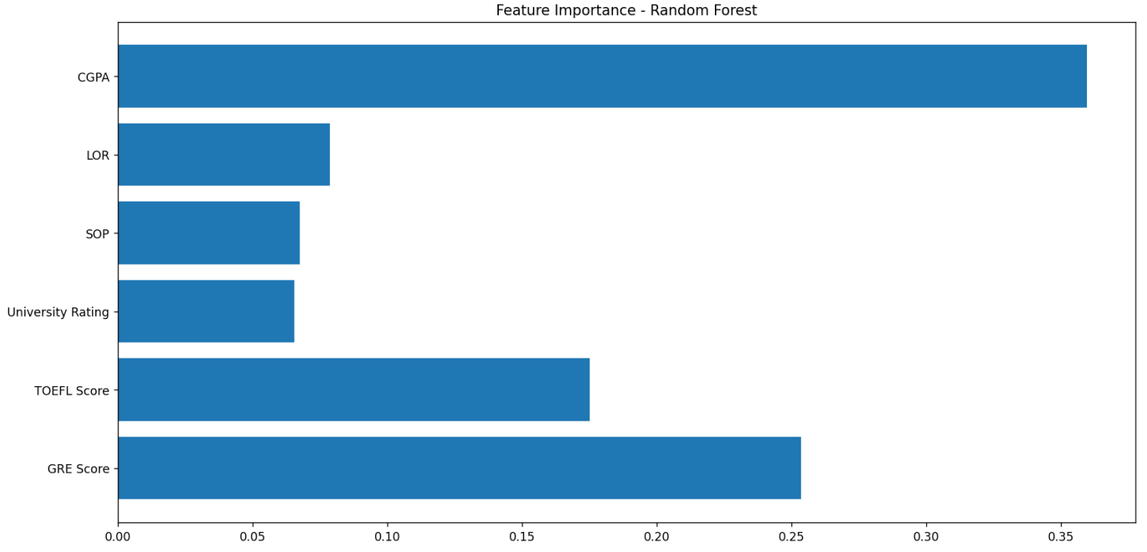
```
import matplotlib.pyplot as plt

plt.barh(X_train.columns,
         tree_model.feature_importances_)
plt.title('Feature Importance - Decision Tree')
plt.show()
```

```
import matplotlib.pyplot as plt

plt.barh(X_train.columns,
         forest_model.feature_importances_)
plt.title('Feature Importance - Random Forest')
plt.show()
```





Let's practice!

EXPLAINABLE AI IN PYTHON