BITS Pilani presentation

BITS Pilani
Pilani Campus

Tanmay Tulsidas Verlekar
CSIS

innovate achieve lead

BITS Pilani

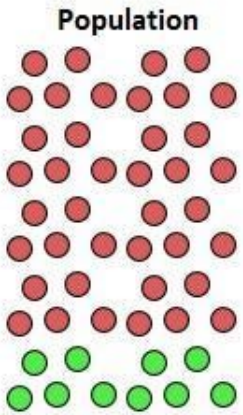
Pilani Campus

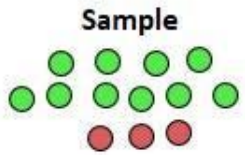
Applied Machine Learning SE ZG568 / SS ZG568

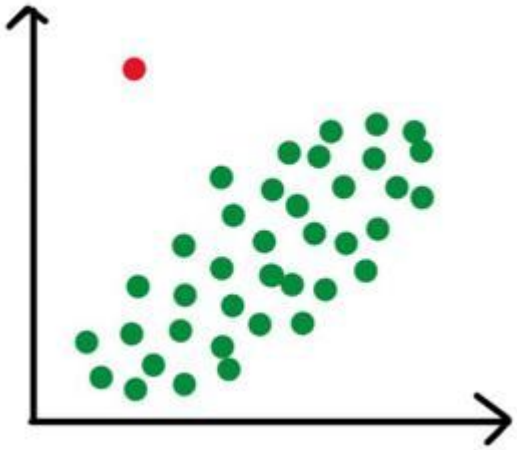
Lecture No. 2

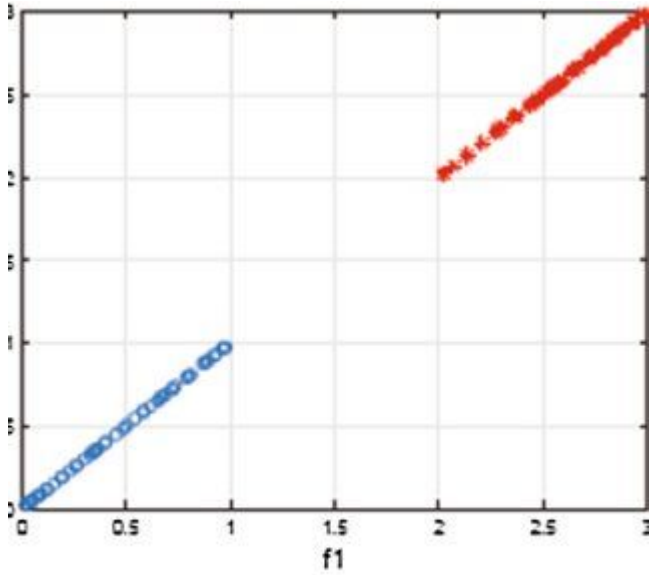
Main Challenges of Machine Learning

- Insufficient Quantity of Training Data
- Nonrepresentative Training Data
- Poor-Quality Data - cleaning data is important
- Irrelevant Features



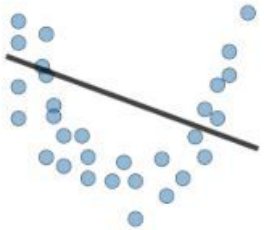
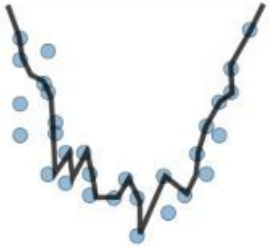
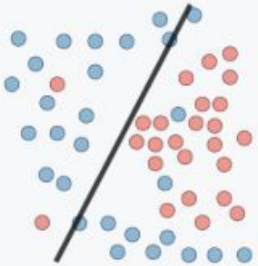
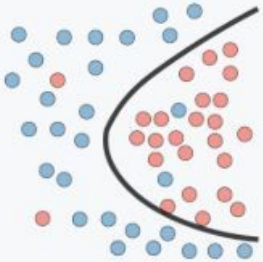
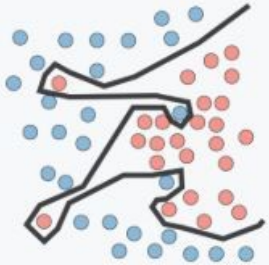
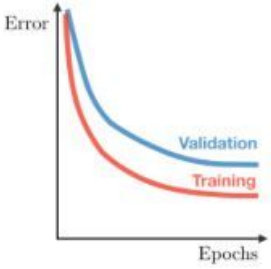
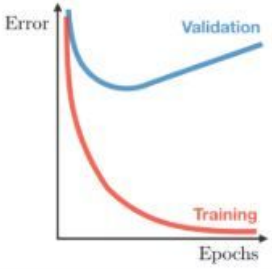




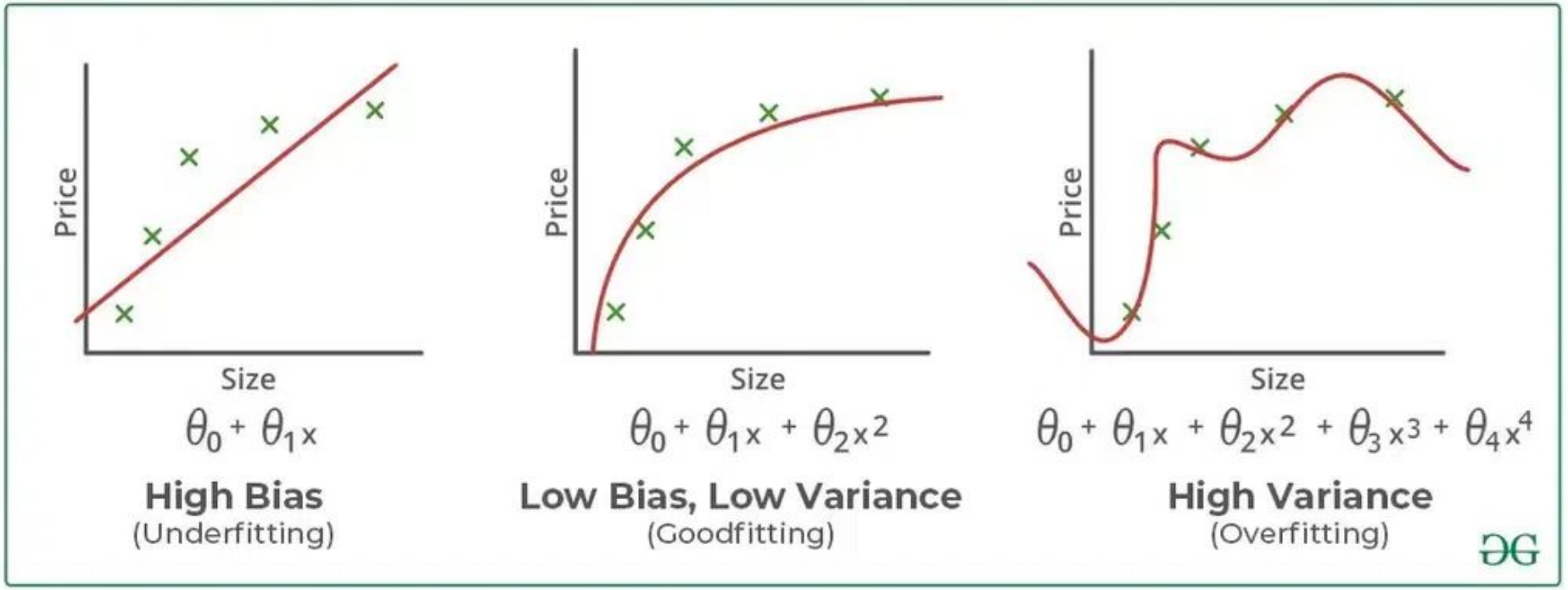


(b) f_1 with f_2

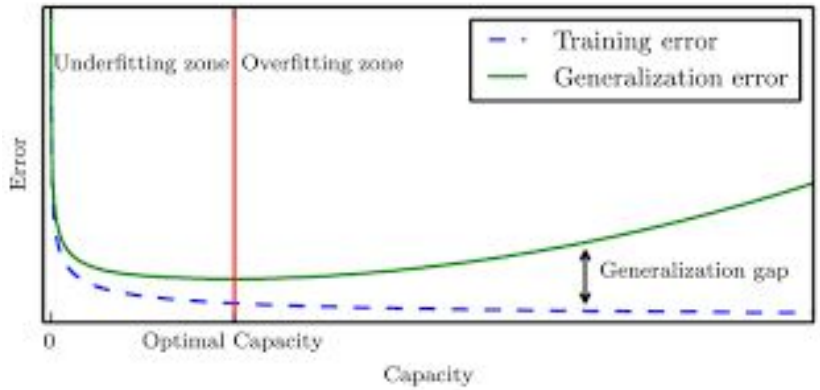
Overfitting and Underfitting

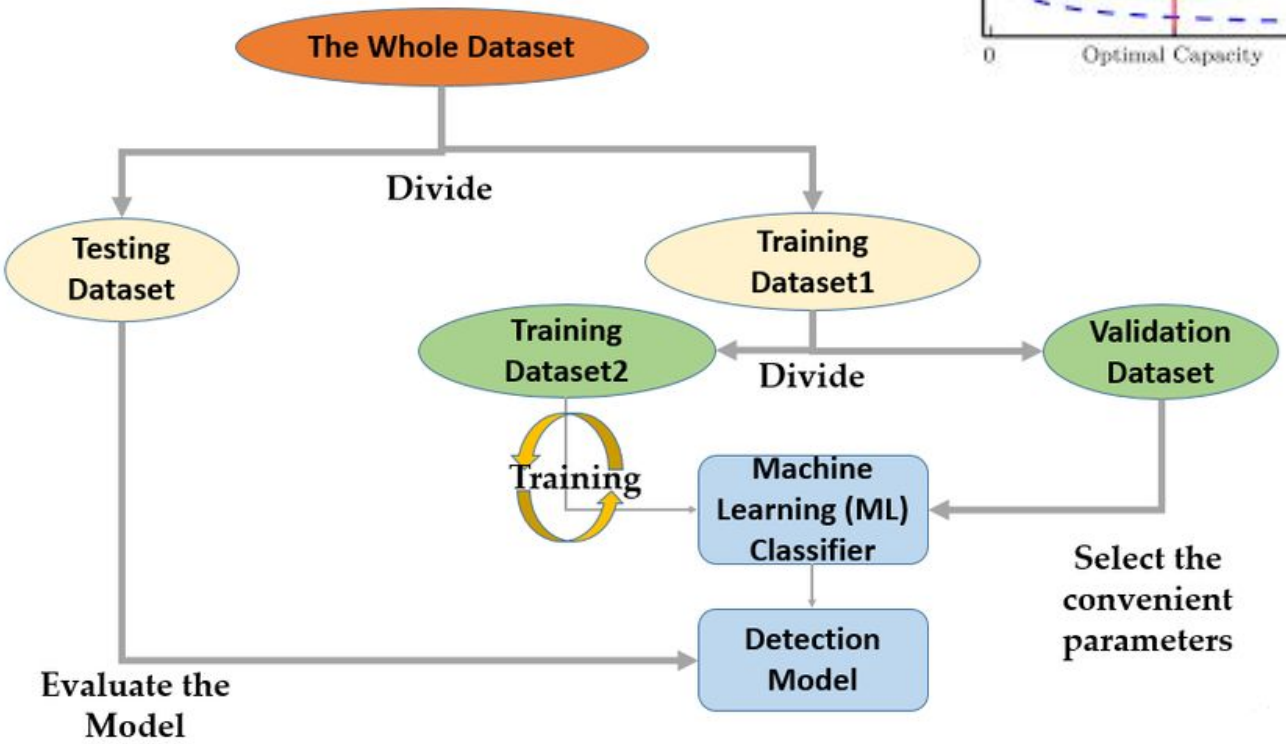
	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

Regularisation

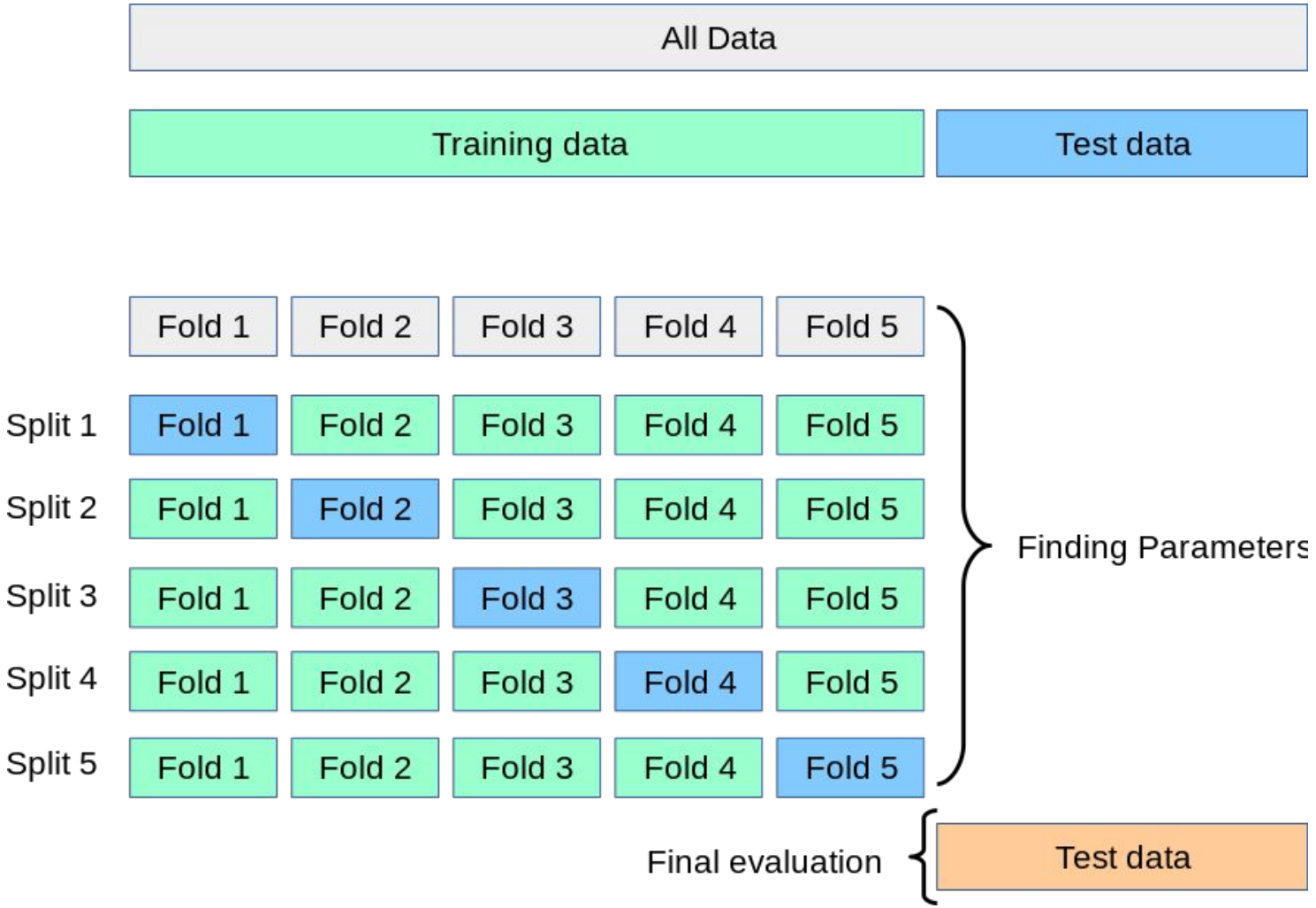


Testing and Validating



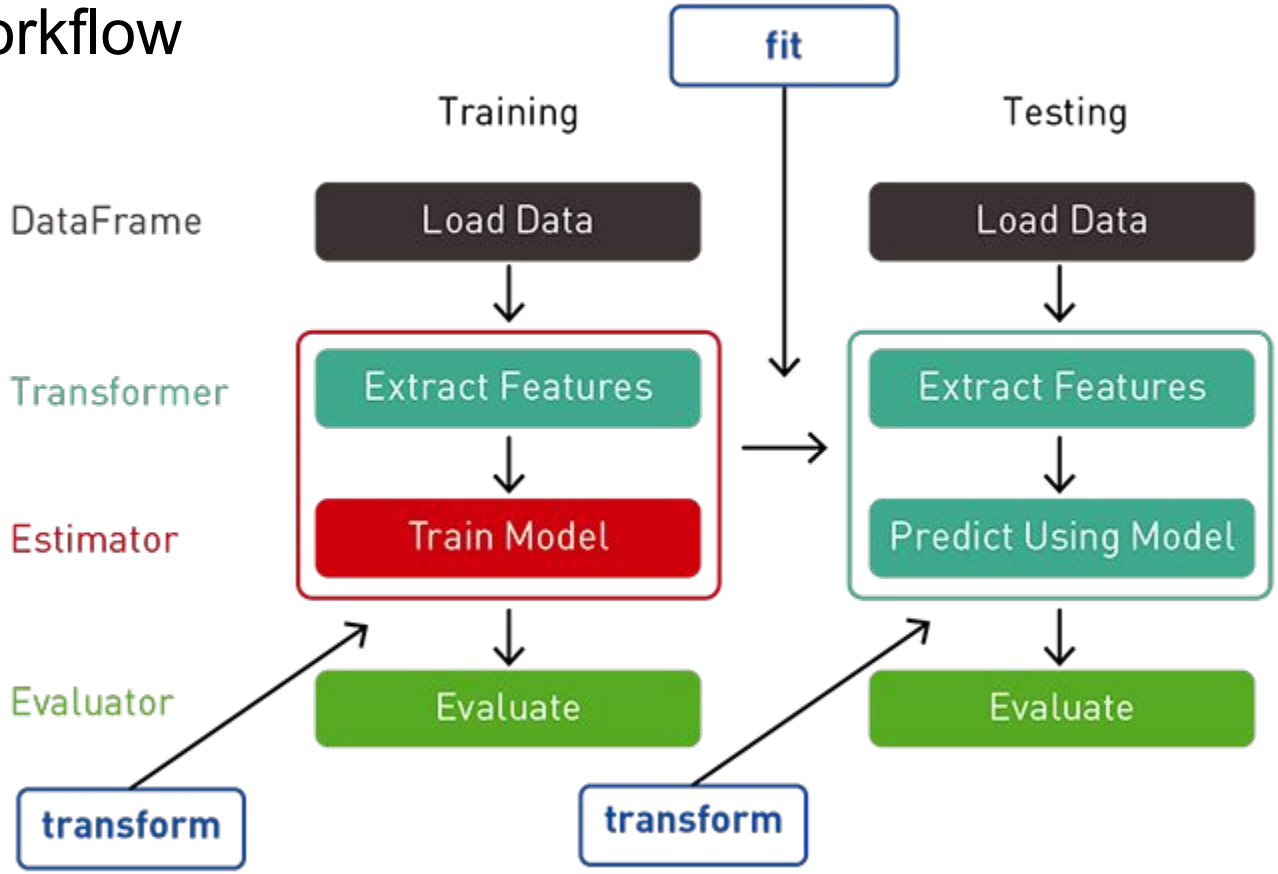


Cross- validation



End-to-End Machine Learning

Simple ML workflow



Look at the Big Picture

Understand the objective.

Review what exists.

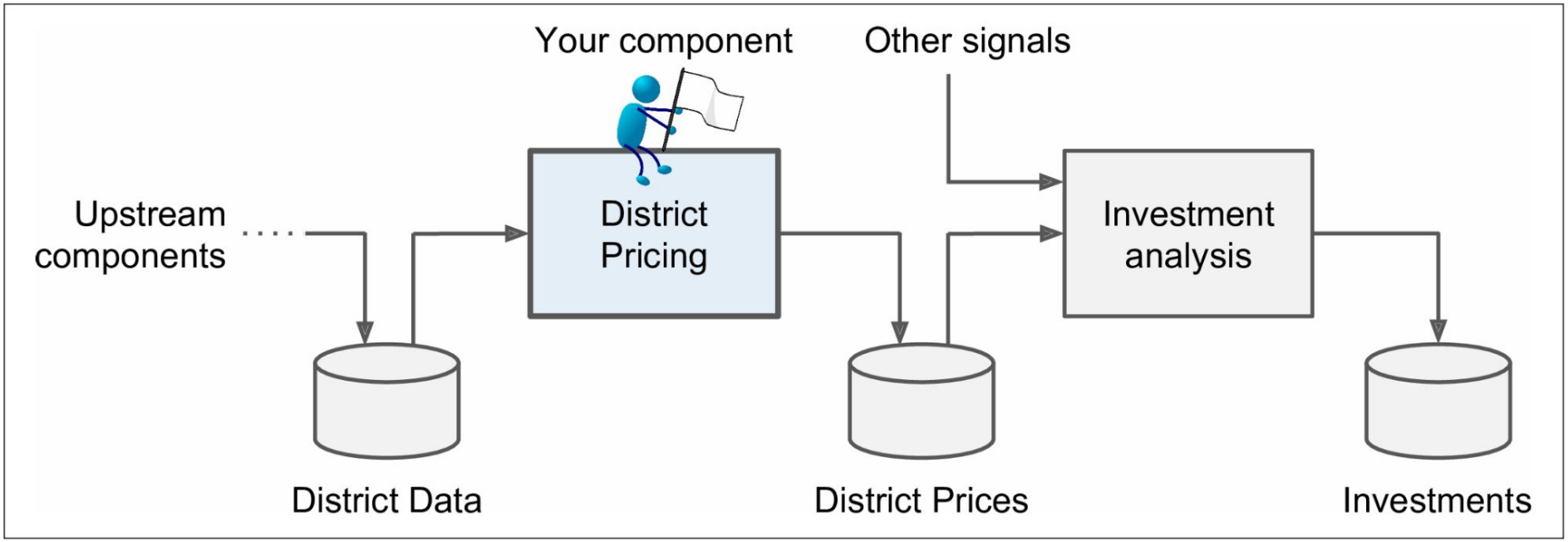


Figure 2-2. A Machine Learning pipeline for real estate investments

Designing your system

is it supervised, unsupervised, or Reinforce

ment Learning?

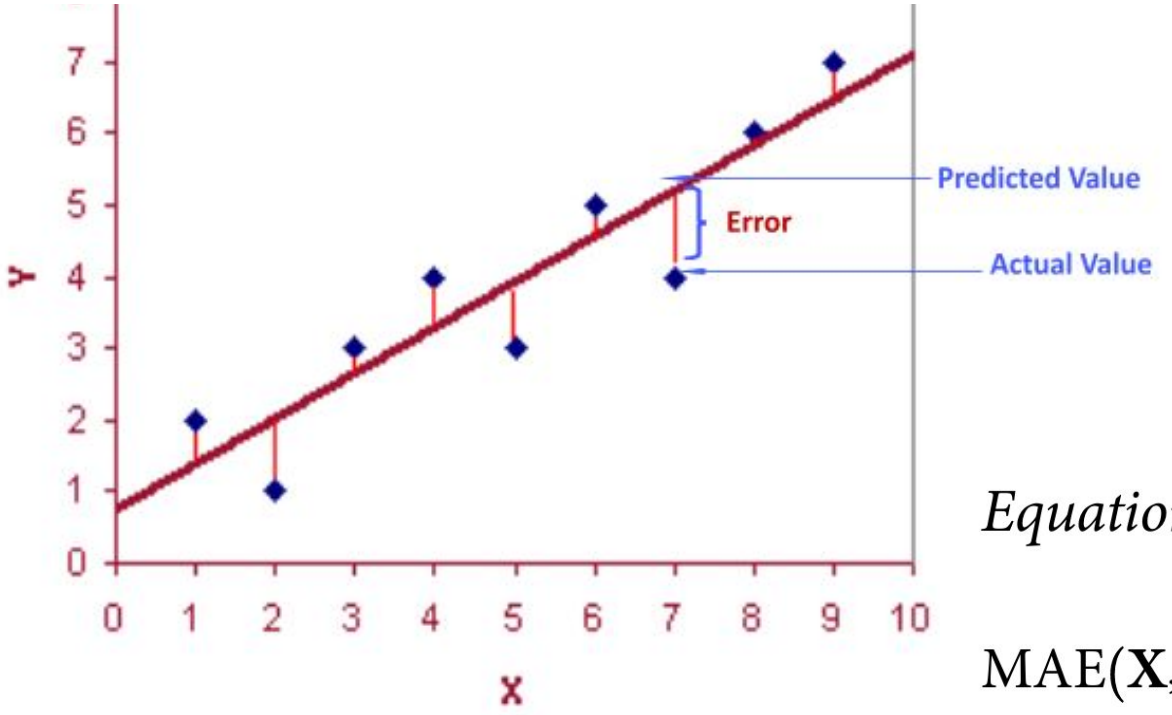
Is it a classification task, a regression task, or something else?

Should you use batch learning or online learning techniques?

Select a Performance Measure

Equation 2-1. Root Mean Square Error (RMSE)

$$\text{RMSE}(\mathbf{X}, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right)^2}$$



Equation 2-2. Mean Absolute Error

$$\text{MAE}(\mathbf{X}, h) = \frac{1}{m}\sum_{i=1}^{m}\left|h\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right|$$

Write your first code

Imports


```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

Dataset

innovate achieve lead

```
data=pd.read_csv('/kaggle/input/diabetesdataanslysis/diabetes.csv')
data.head(10)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

Try .info(), .describe()

Split data

```
X=data.drop("Outcome",axis=1)
y=data['Outcome']
```

```
#Splitting the data into data train and data test
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=42
```

The ML model

```
#Creating model using data train
model=LogisticRegression(solver='lbfgs', max_iter=1000)
model.fit(X_train,y_train)
```

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

Analysis

```
# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print('Mean squared error: %.2f'
      % mean_squared_error(diabetes_y_test, diabetes_y_pred))
# The coefficient of determination: 1 is perfect prediction
print('Coefficient of determination: %.2f'
      % r2_score(diabetes_y_test, diabetes_y_pred))
```

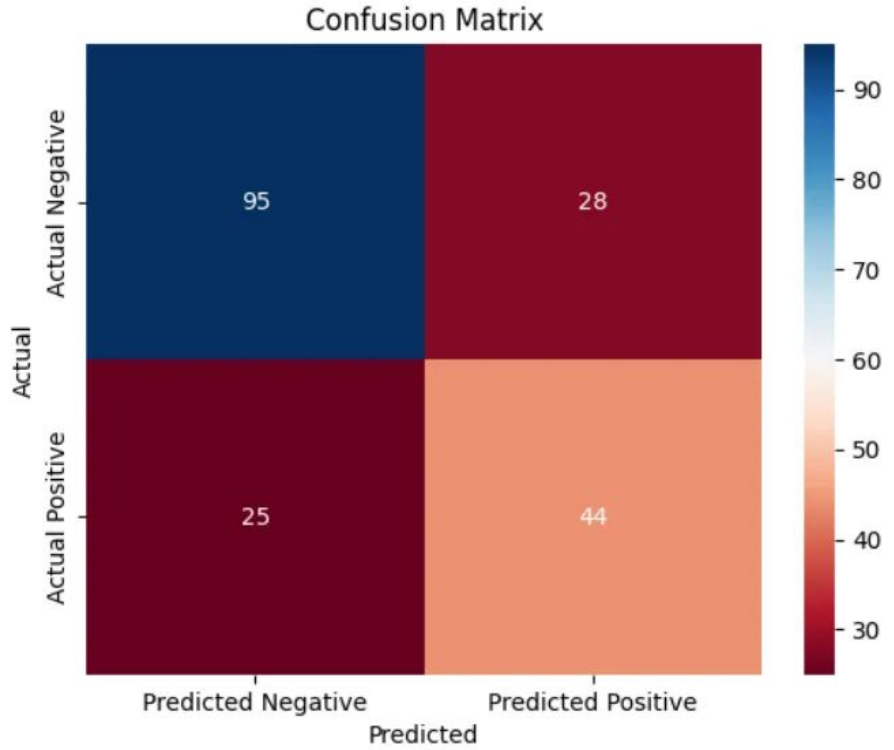
```
#Accuracy
accuracy= accuracy_score(y_test,y_pred)
print('Accuracy:',round(accuracy,2))

#Classification Report
print(classification_report(y_test,y_pred))

#Confussion matrix
conf_matrix=confusion_matrix(y_test,y_pred)
print(conf_matrix)
```

Visualition

```
sns.heatmap(conf_matrix,cmap='RdBu',annot=True,yticklabels=['Actual Negative'
            xticklabels=['Predicted Negative','Predicted Positive'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



Visualition

```
#Scatter Plot
plt.scatter(diabetes_X_test, diabetes_y_test,  color='black')
plt.plot(diabetes_X_test, diabetes_y_pred, color='blue', linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```