Recap

- Polynomial Regression
- Bias, variance, and the tradeoff between them
- Overfitting and underfitting
- Regularization: Lasso, Ridge, Elastic Net
- Implementing regularization in python
- Cross Validation: K-Fold, Leave-One-Out, Three-Way Data Split

Week 6: Data Science Part-Time Course

KNN/Classification

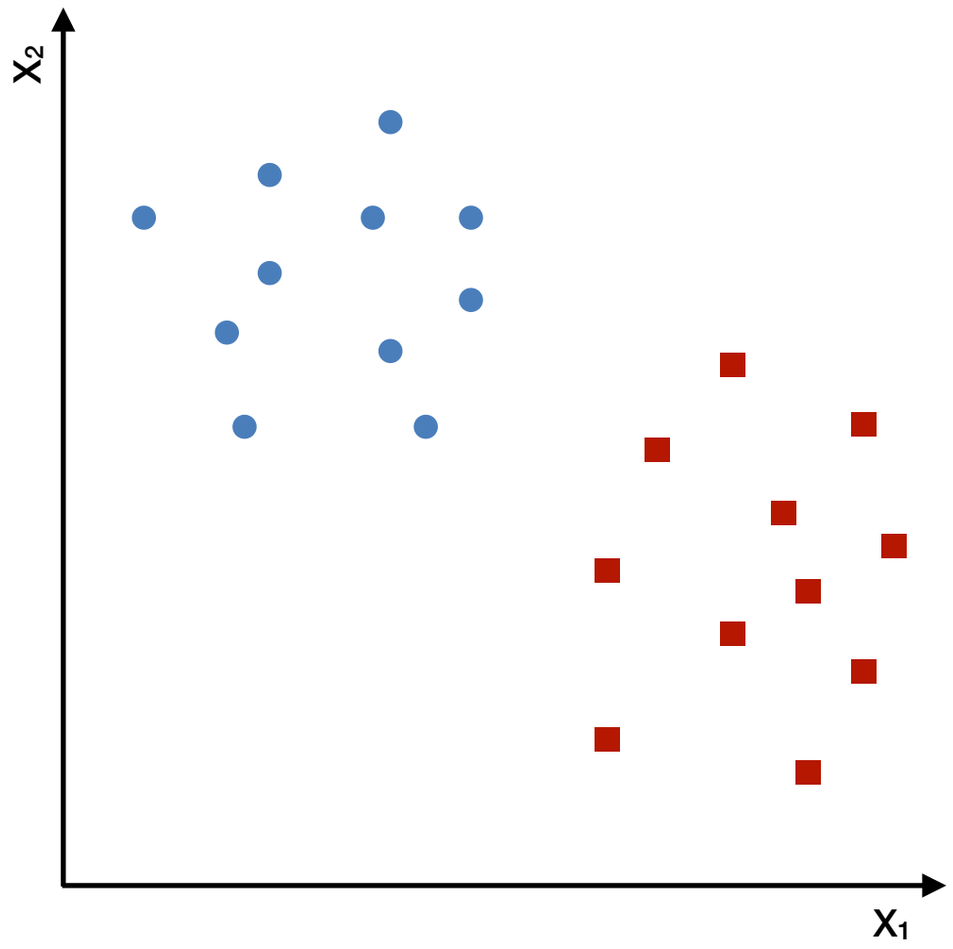
Dami Lasisi

Intro to Classification

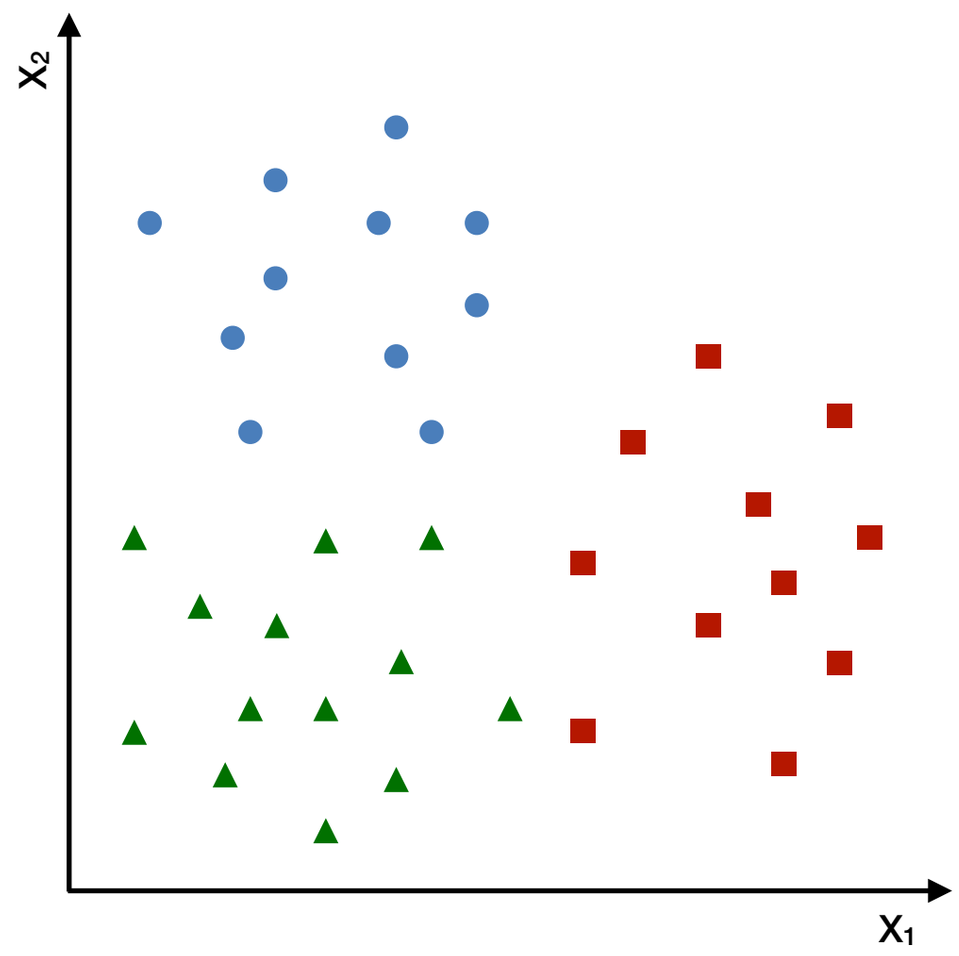
- Supervised Learning models
- Draws a conclusion from observed values
- Often used to predict binary outcomes, but some classification problems are not binary
- Outcomes are class labels that can be applied to the dataset (e.g. using symptoms and medical history to predict whether or not a patient has cancer
- Examples of classification models:
  - K-Nearest Neighbours (KNN)
  - Logistic Regression
  - Tree-based models

Examples of class labels:
- **cancer, no cancer**
- **cat, dog, rabbit**
- **default, not default**

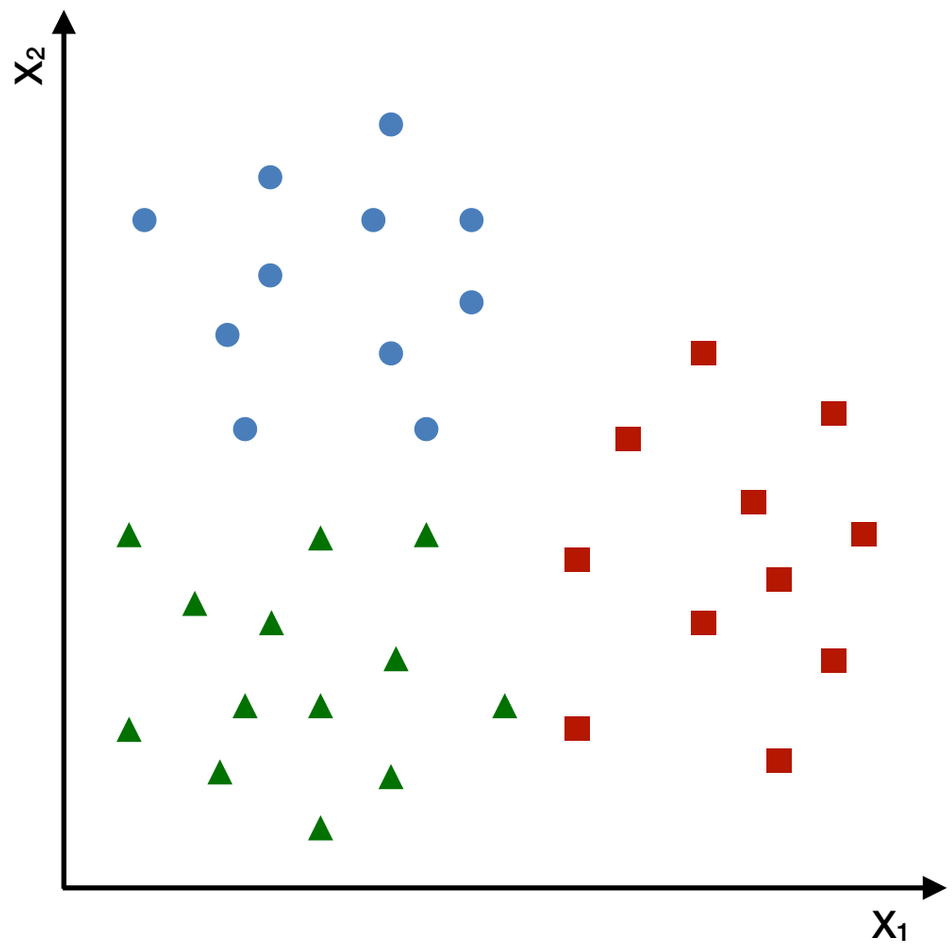
Binary classification

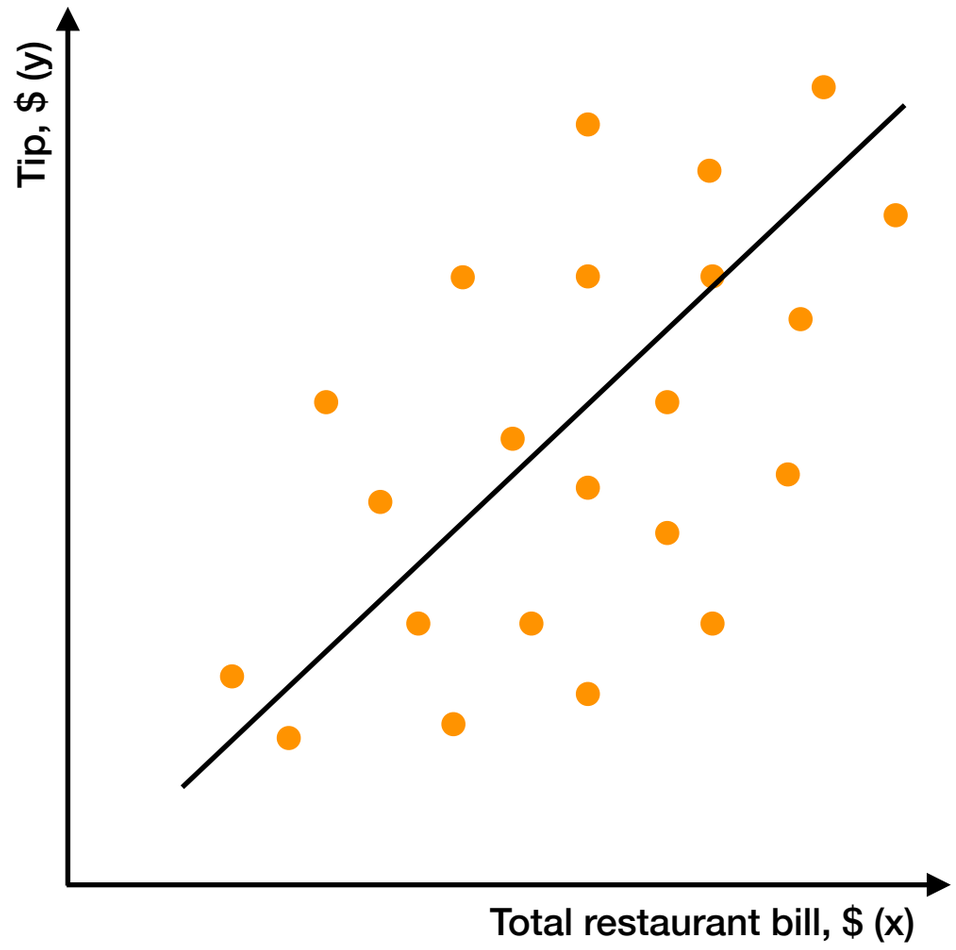


Multi-class classification



Classification vs Regression





Determining which model to pick: Regression or Classification

- What does the target (dependent) variable look like?
- Can the target variable be ordered mathematically?
- For example, if predicting waiters' tip , $20 is greater than $18. This is a regression problem because the target can be ordered.
- If we're predicting cancer diagnosis, having cancer is not inherently greater than not having cancer. This is a classification problem.
- Note: class labels in a dataset do not overlap.

Activity: Regression or Classification

- Using the total number of explosions in a movie, predict if the movie is by JJ Abrams or Michael Bay.
- Determine how many tickets will be sold to a concert given who is performing, where, and the date and time.
- Given the temperature over the last year by day, predict tomorrow's temperature outside.
- Using data from four cell phone microphones, reduce the noisy sounds so the voice is crystal clear to the receiving phone.
- With customer data, determine if a user will return or not in the next 7 days to an e-commerce website.

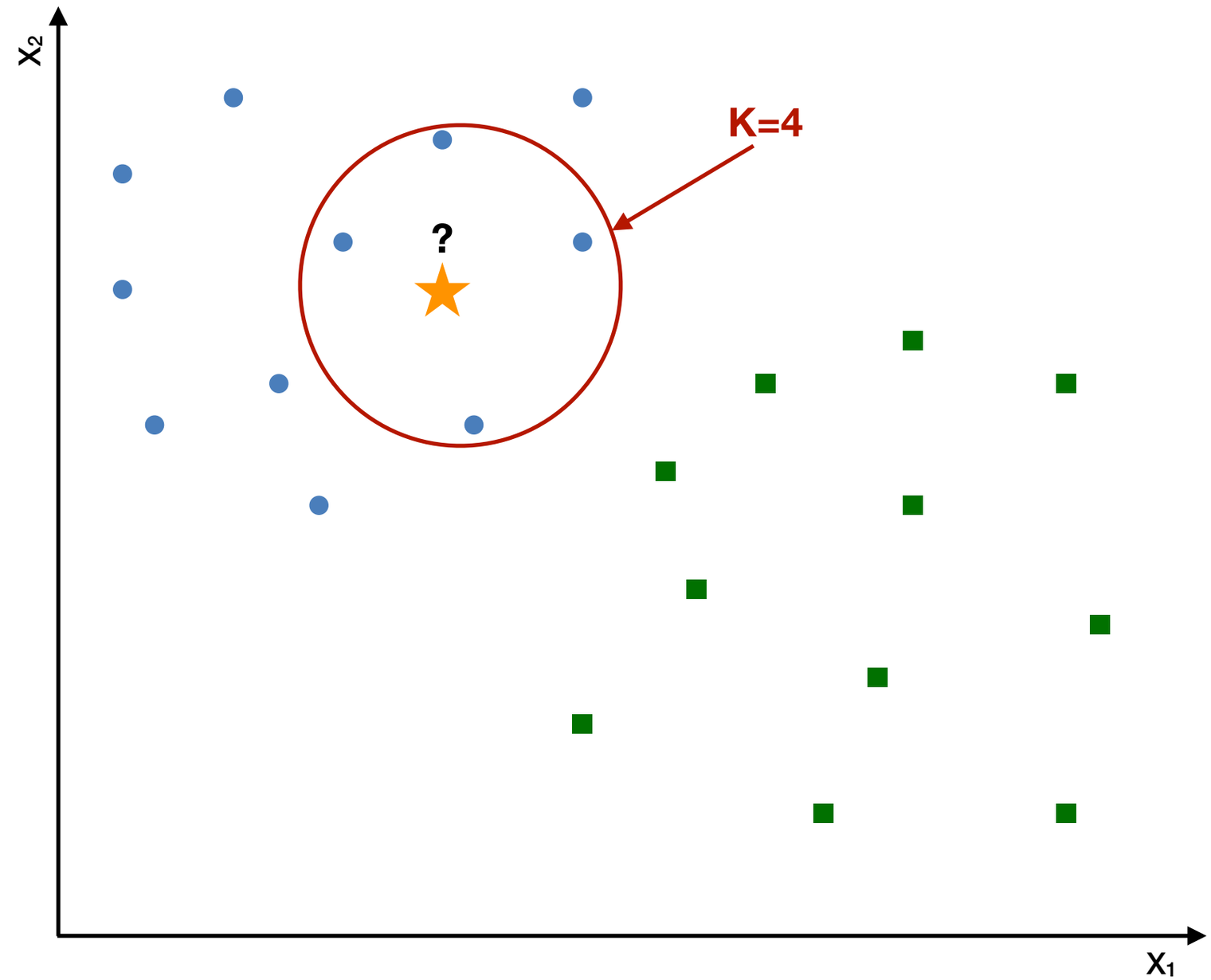
KNN

- Classification algorithm that make predictions based on the observations closest to the one we're trying predict.

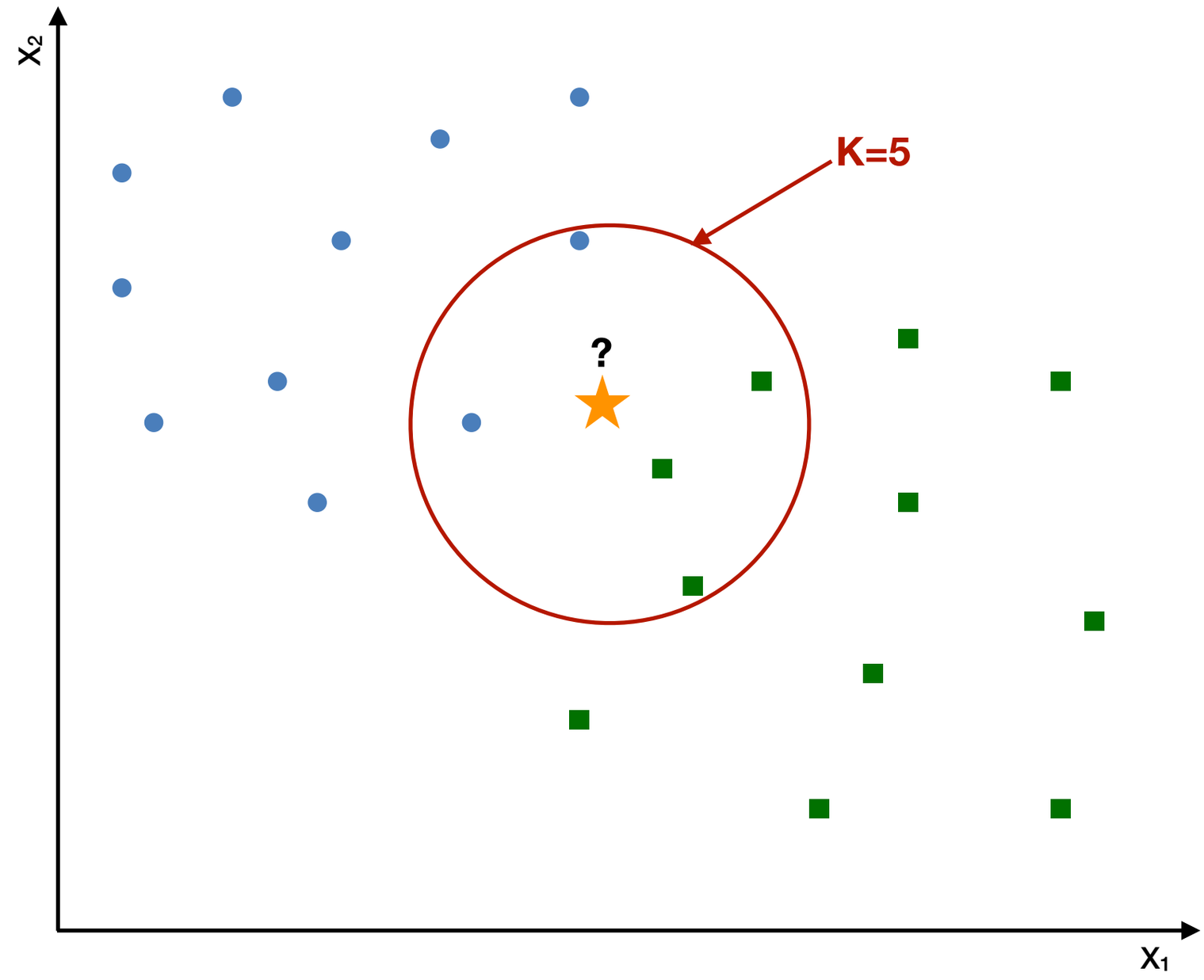
- Non-parametric: no assumptions or data distribution requirements

- How the algorithm works:
  - For a given point, calculate the distance to all other points.
  - Given those distances, pick the *k* closest points.
  - Calculate the probability of each class label given those points.
  - The original point is classified as the class label with the largest probability ("votes").

KNN: How the algorithm works



KNN: How the algorithm works



Implementing KNN in Python with Scikit-Learn

In [1]:
```
from sklearn import datasets, neighbors
import pandas as pd
import numpy as np
```

In [2]:
```
iris = datasets.load_iris()
df_iris = pd.DataFrame(data= np.c_[iris['data'], iris['target']],
                       columns= iris['feature_names'] + ['target'])
df_iris['species'] = pd.Categorical.from_codes(iris.target, iris.target_names)
df_iris.head()
```

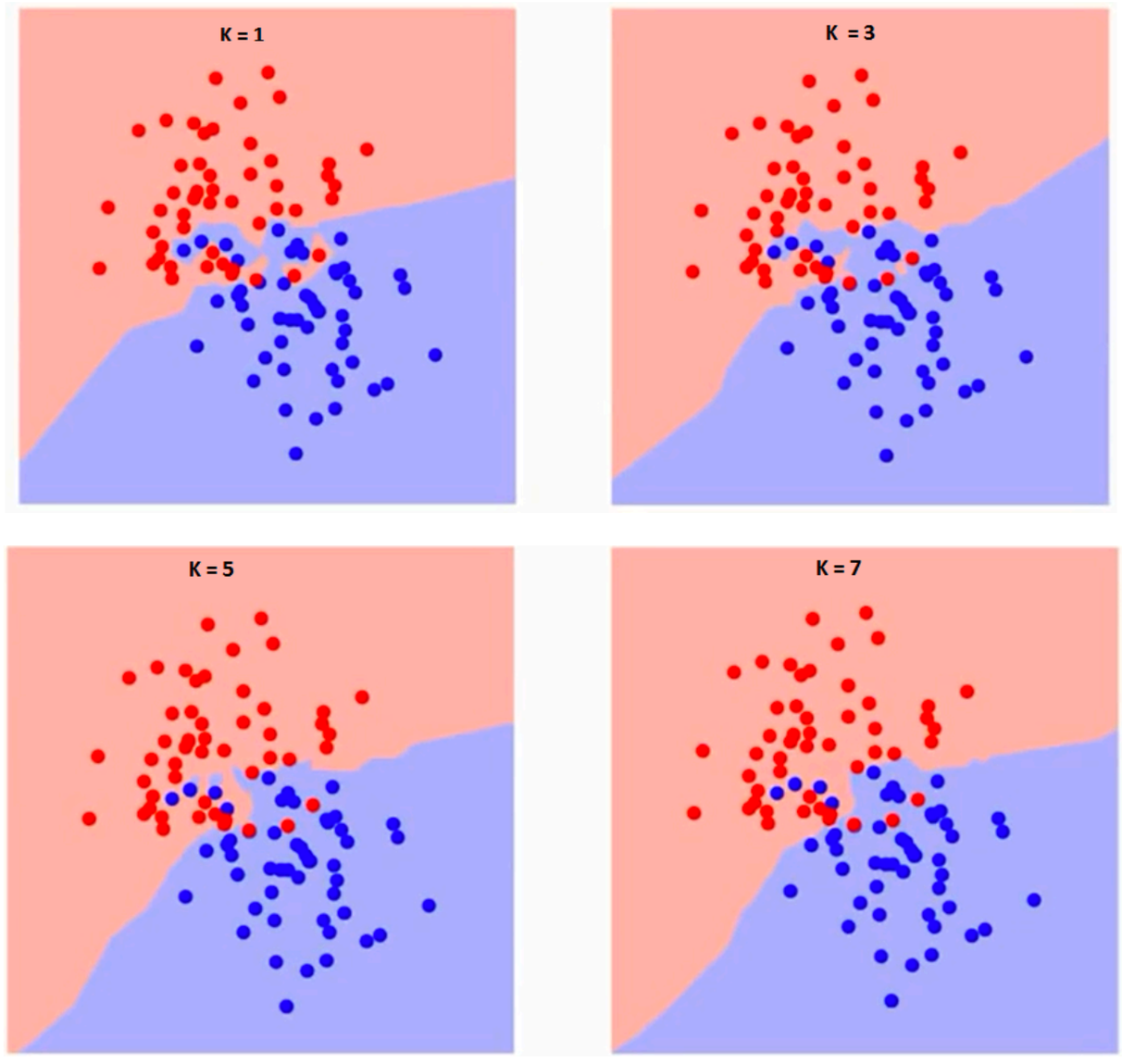
Out[2]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target	species
0	5.1	3.5	1.4	0.2	0.0	setosa
1	4.9	3.0	1.4	0.2	0.0	setosa
2	4.7	3.2	1.3	0.2	0.0	setosa
3	4.6	3.1	1.5	0.2	0.0	setosa
4	5.0	3.6	1.4	0.2	0.0	setosa

In [3]:
```
knn = neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform')
knn.fit(iris.data[:,2:], iris.target)
print (knn.predict(iris.data[:,2:]))
print (knn.score(iris.data[:,2:], iris.target))
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1
 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 1 2 2 2 2
 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
0.96
```

KNN: How do we choose K?



KNN: GridSearch

In [5]:
```
from sklearn.model_selection import GridSearchCV
```

In [10]:
```
params = {'n_neighbors': [3,4,5],
          'weights':['uniform', 'distance']}
gs = GridSearchCV(estimator=neighbors.KNeighborsClassifier(), param_grid=params, cv=5)
gs.fit(iris.data, iris.target)
print(gs.best_score_)
print(gs.best_estimator_)
```

```
0.9733333333333334
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=4, p=2,
           weights='uniform')
```

Classification Metrics

- **Accuracy:** the number of *correct* predictions out of all predictions in the sample. This is a value we want to *maximize*.
- **Misclassification rate:** the number of *incorrect* predictions out of all predictions in the sample. This is a value we want to *minimize*
  - 1 - misclassification = accuracy

In [ ]:
```
from sklearn import metrics
y_test = iris.target
y_pred = knn.predict(iris.data[:,2:])

print(metrics.accuracy_score(y_test, y_pred))
```

Advantages and Disadvantages of KNN

- **Advantages:**
  - It's simple to understand and explain.
  - Model training is fast.
  - It can be used for classification and regression (for regression, take the average value of the K nearest points!).
  - Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular.

- **Disadvantages:**
  - It must store all of the training data.
  - Its prediction phase can be slow when n is large.
  - It is sensitive to irrelevant features.
  - It is sensitive to the scale of the data.Accuracy is (generally) not competitive with the best supervised learning methods.