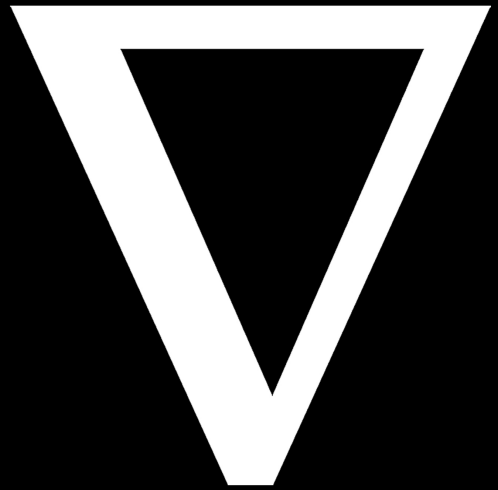


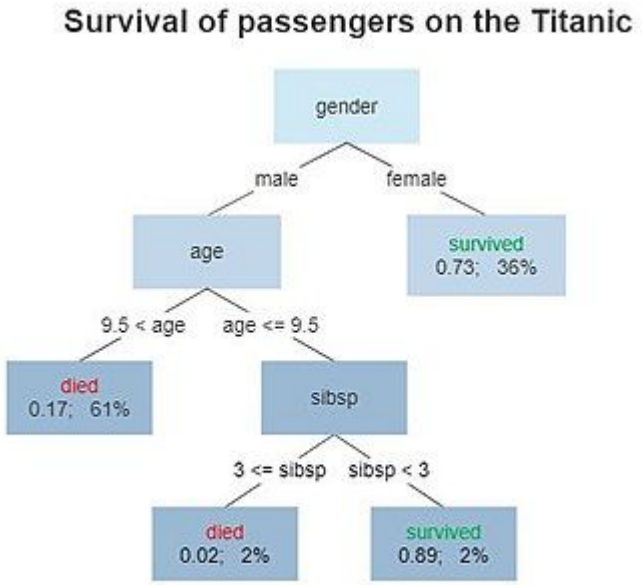
Bagging e Random Forests

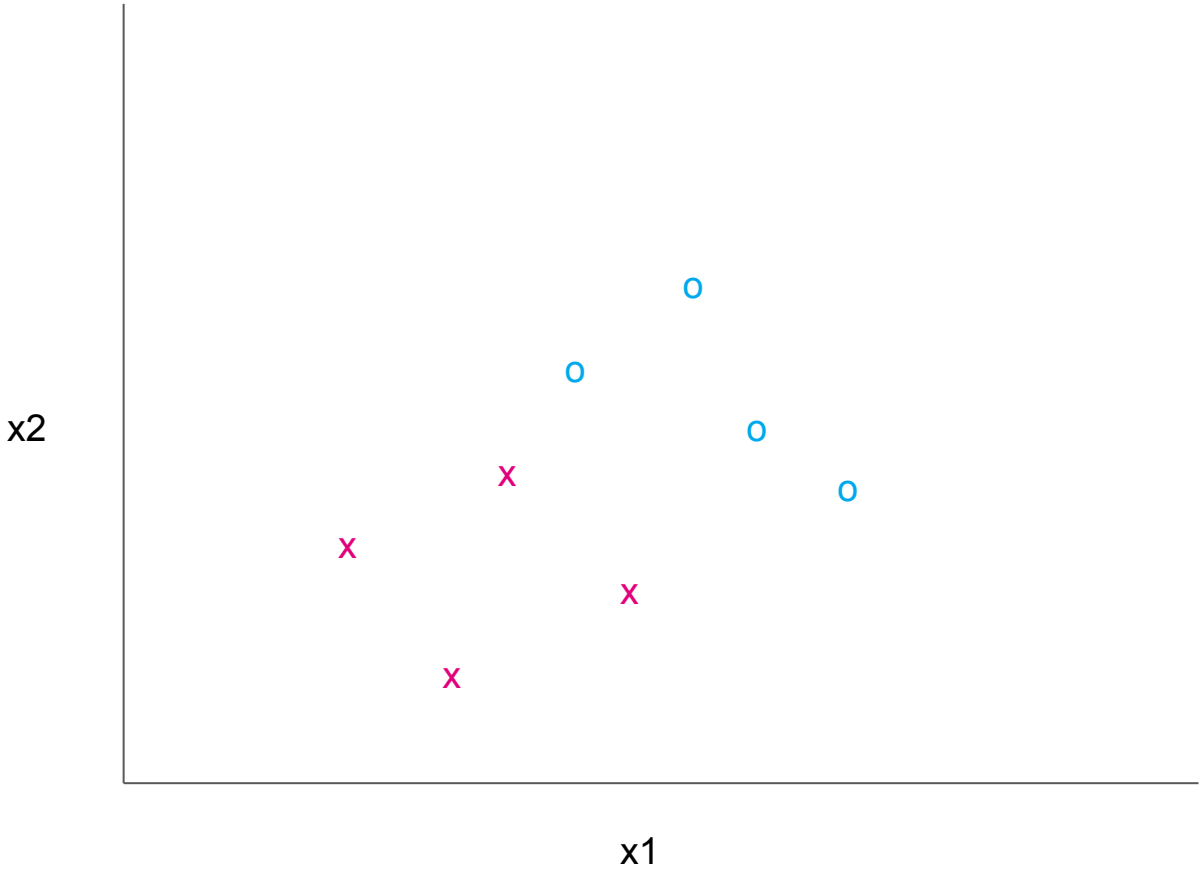
João Marcos Cardoso da Silva
@joaomarcoscsilva

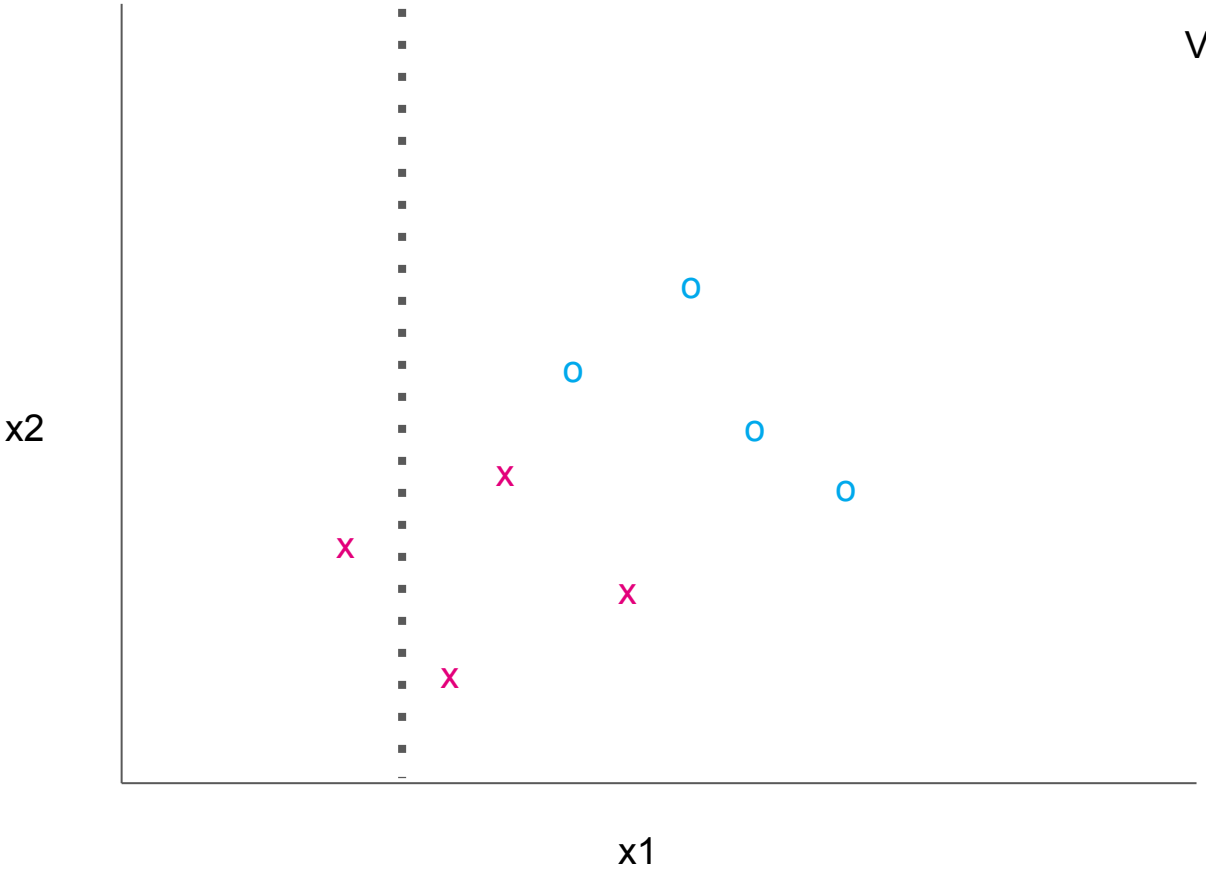
Relembrando Árvores de Decisão

Em árvores de decisão, nós usamos os nós em uma árvore para determinar a categoria a ser predita.

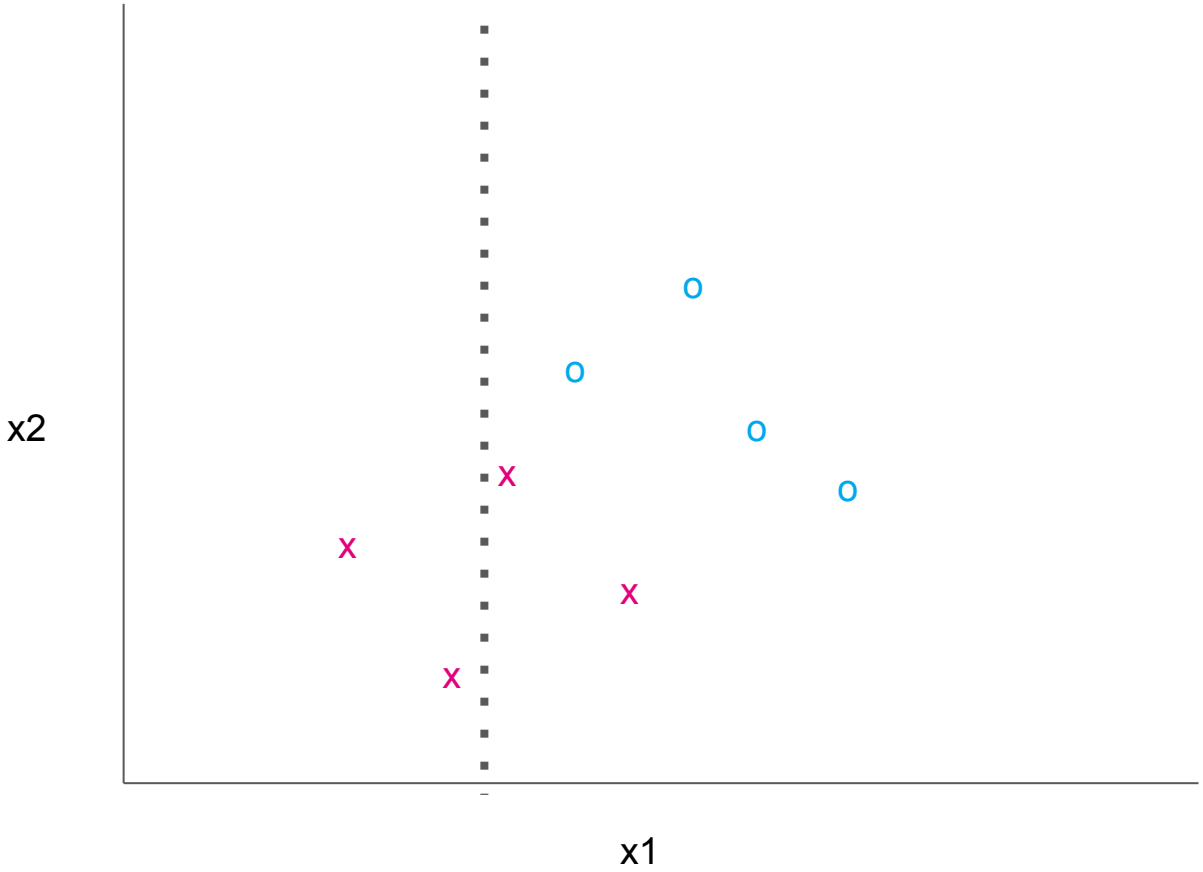
Para fazer predições, nós começamos na raíz e descemos os nós da árvore até chegarmos em uma folha, que determina a classe final.

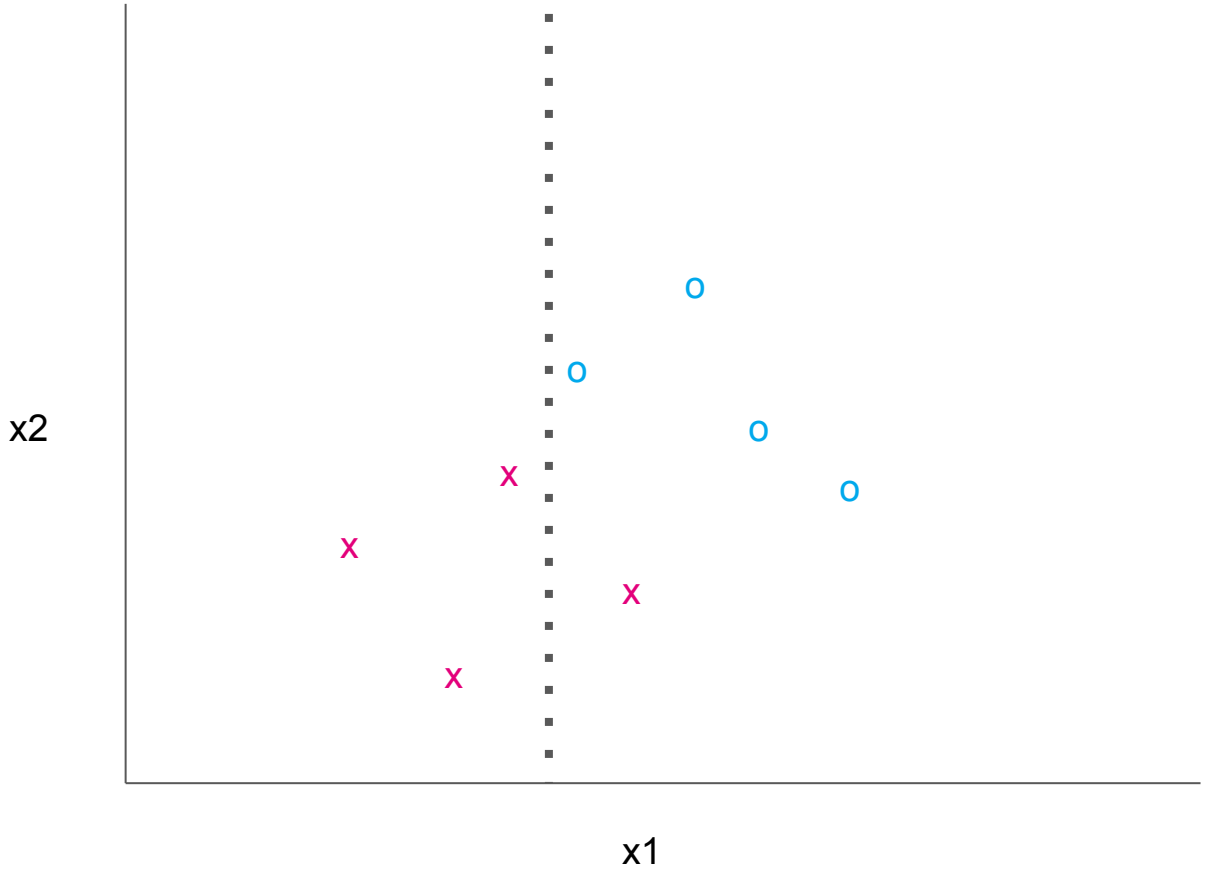


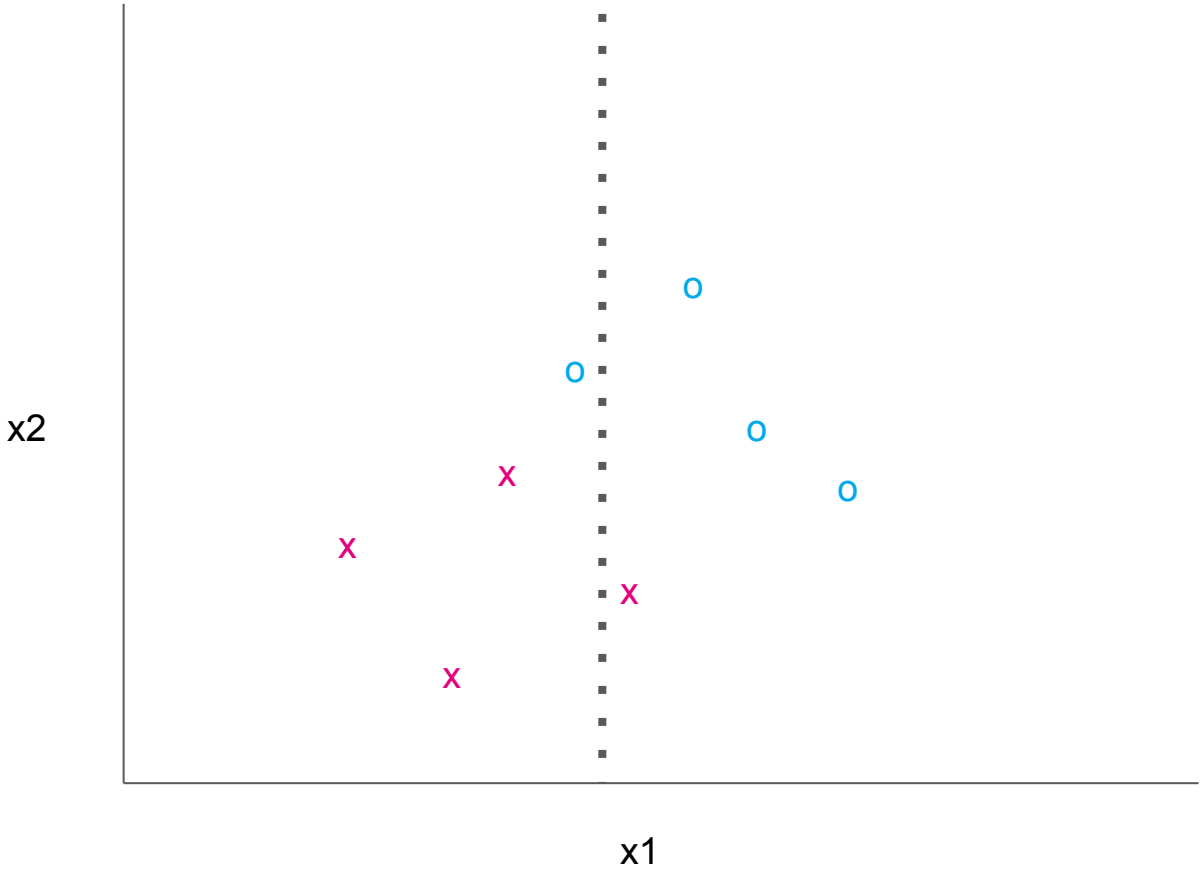

x2
o
o
o
o
x
x
x
x
x1

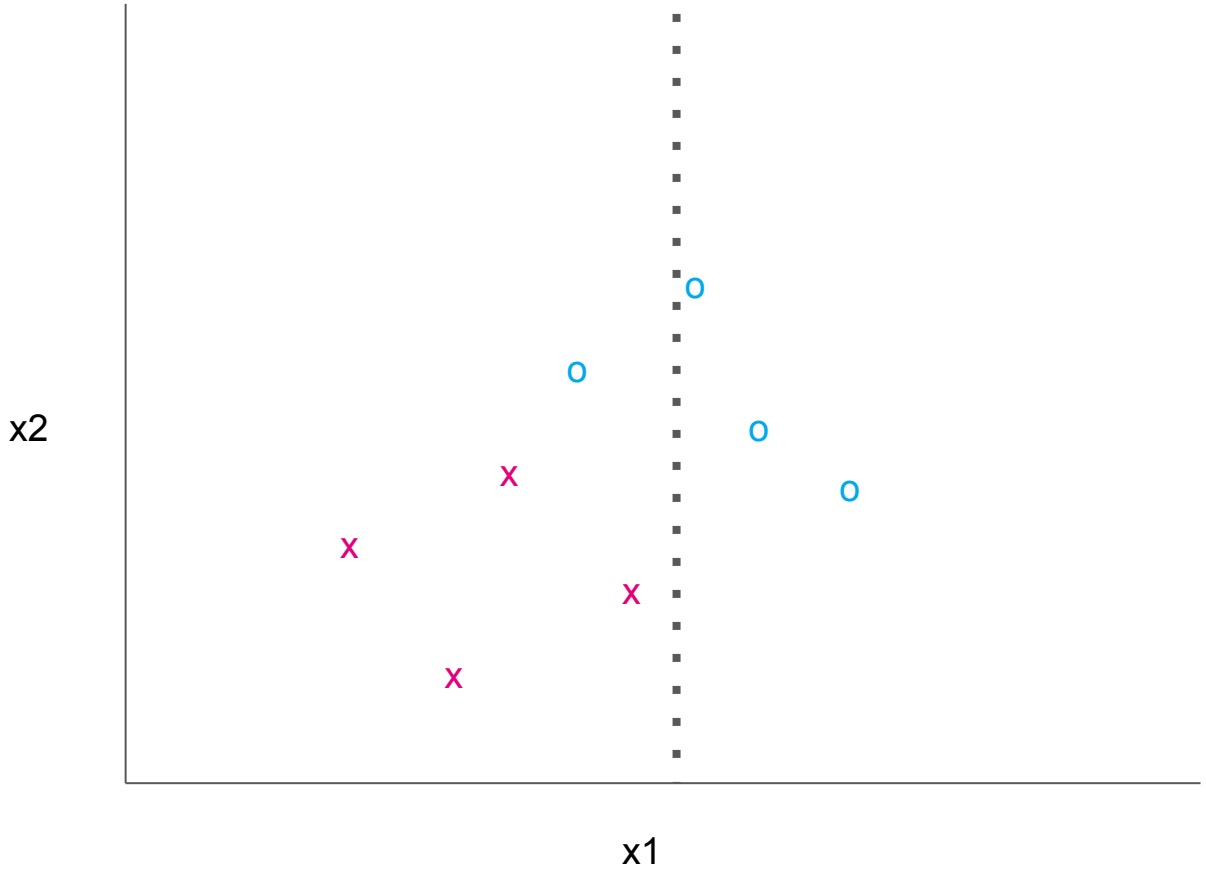


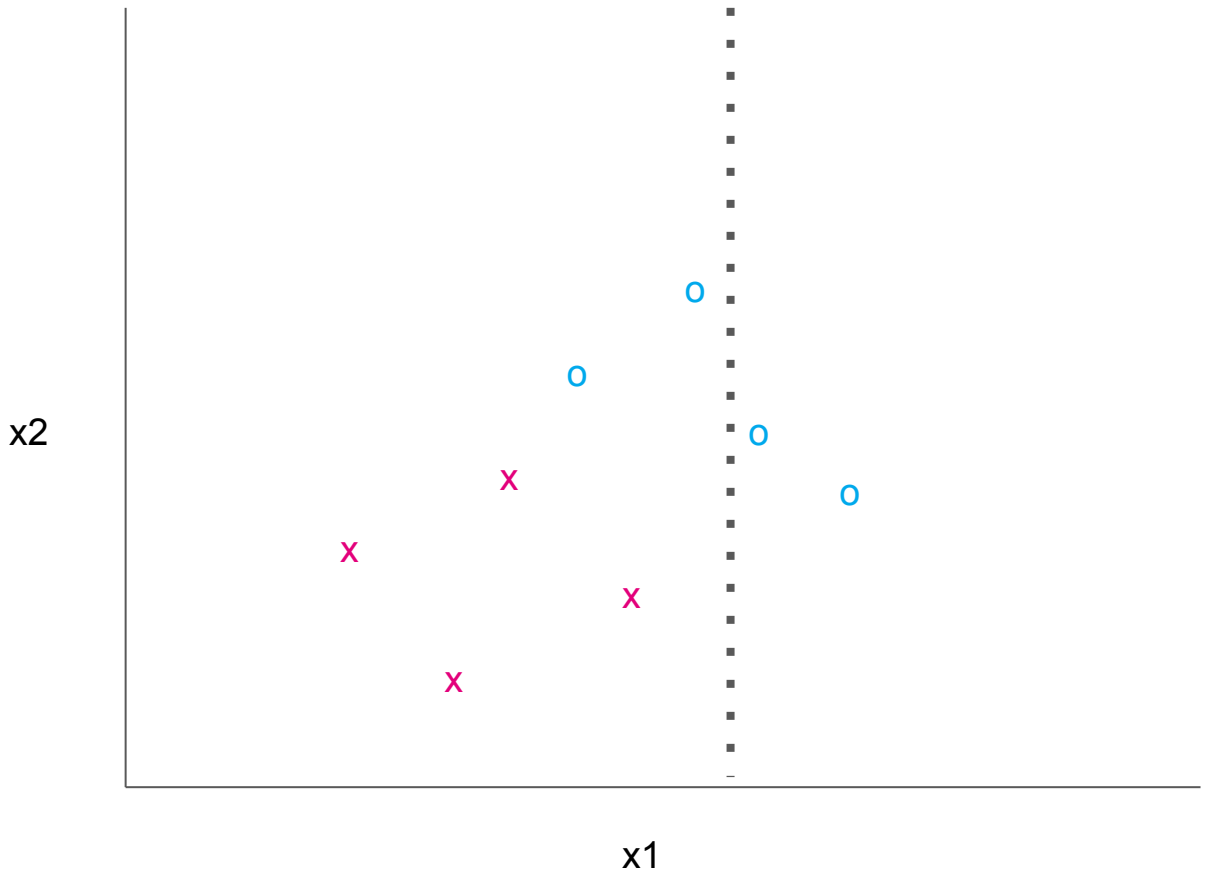
Vamos tentar separar nossos dados pela *feature* x1

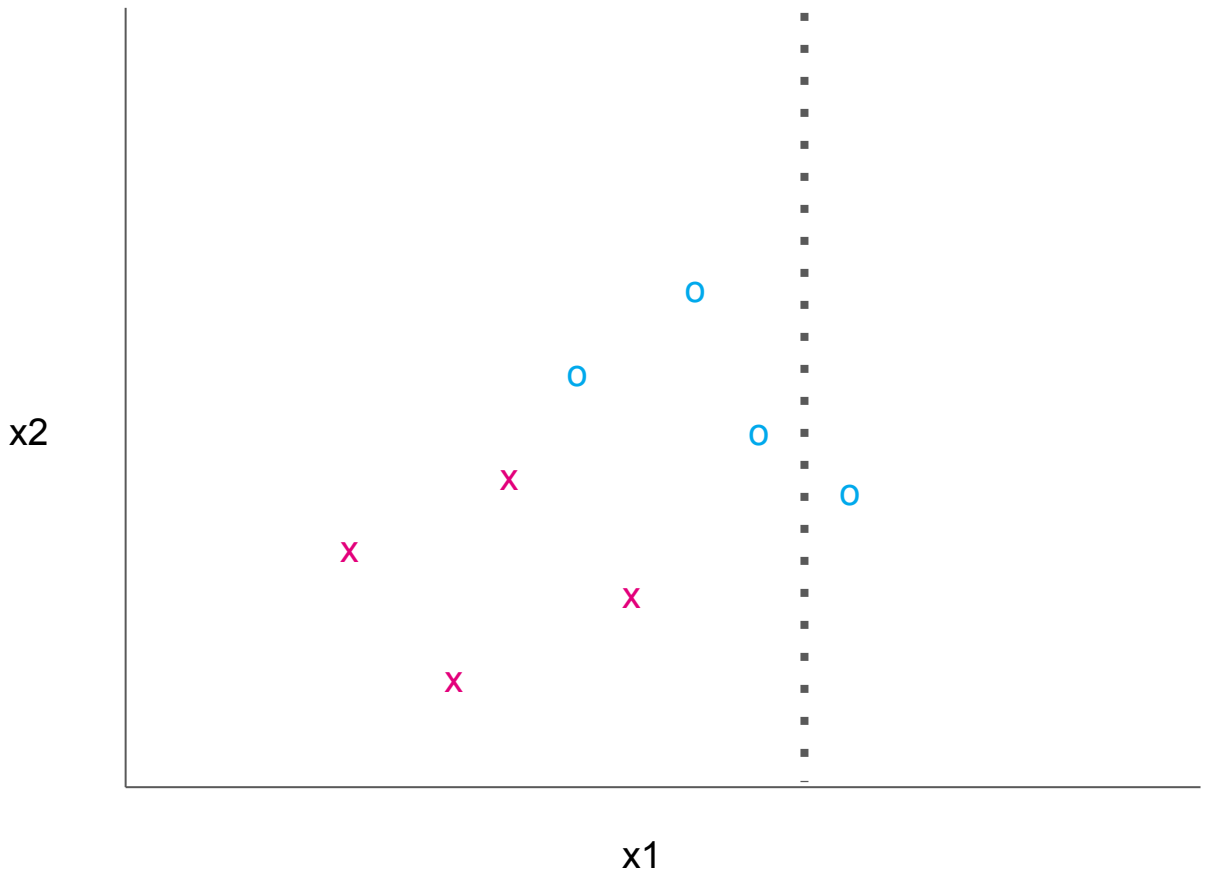

x2
x1
o
o
o
o
x
x
x
x

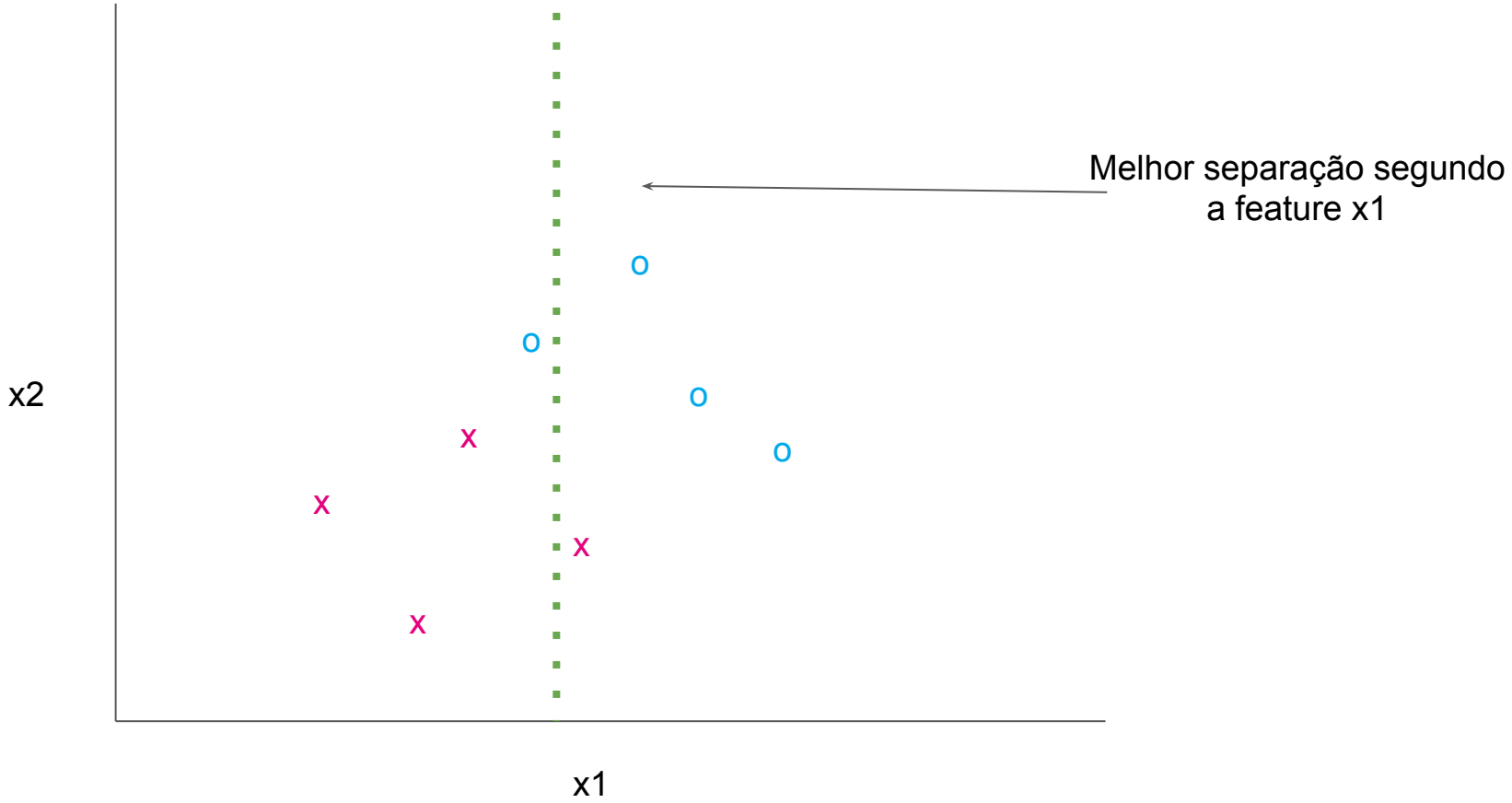

x2
x1
o
o
o
o
x
x
x
x

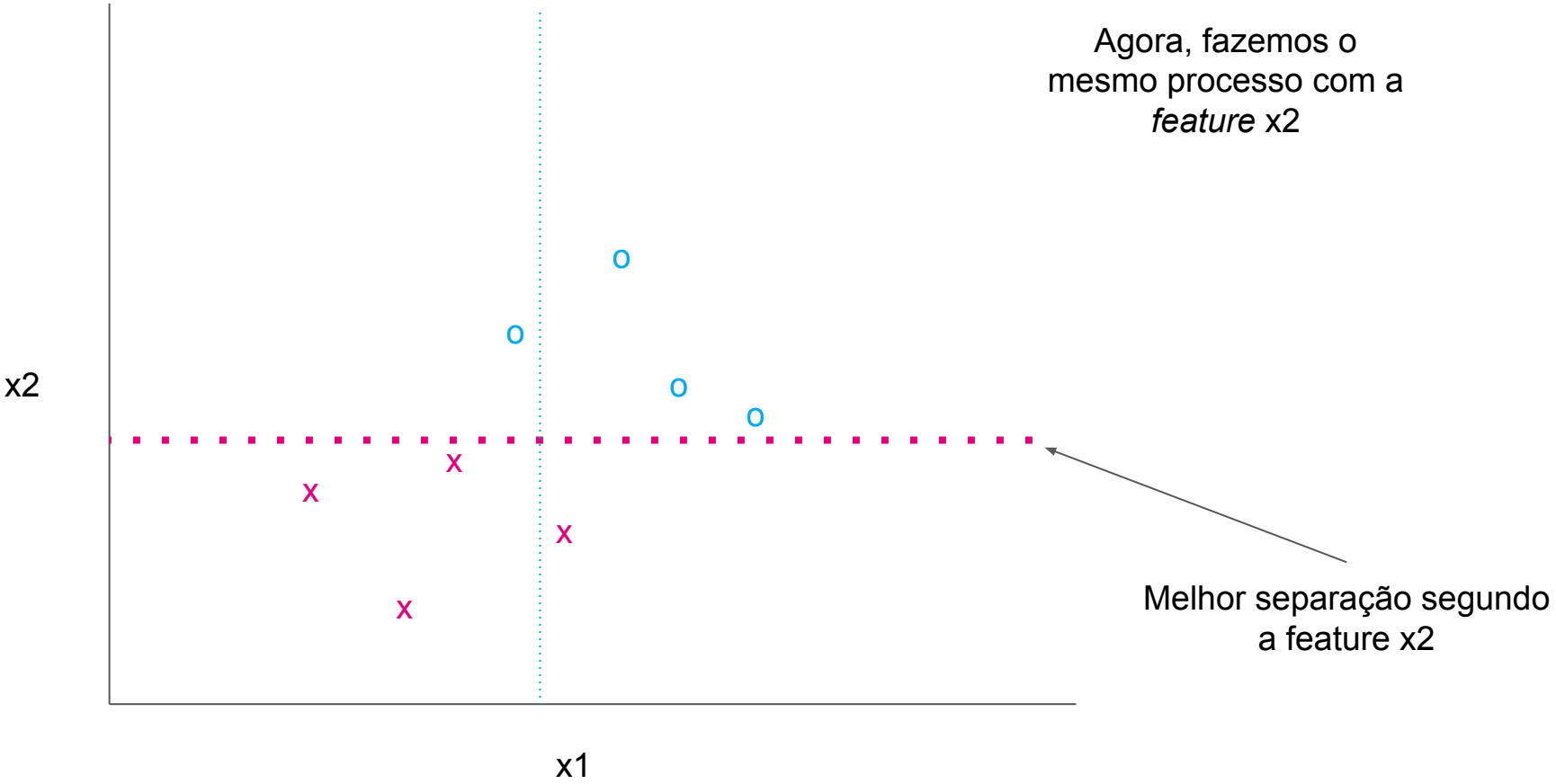

x2
x1
o
o
o
o
x
x
x
x

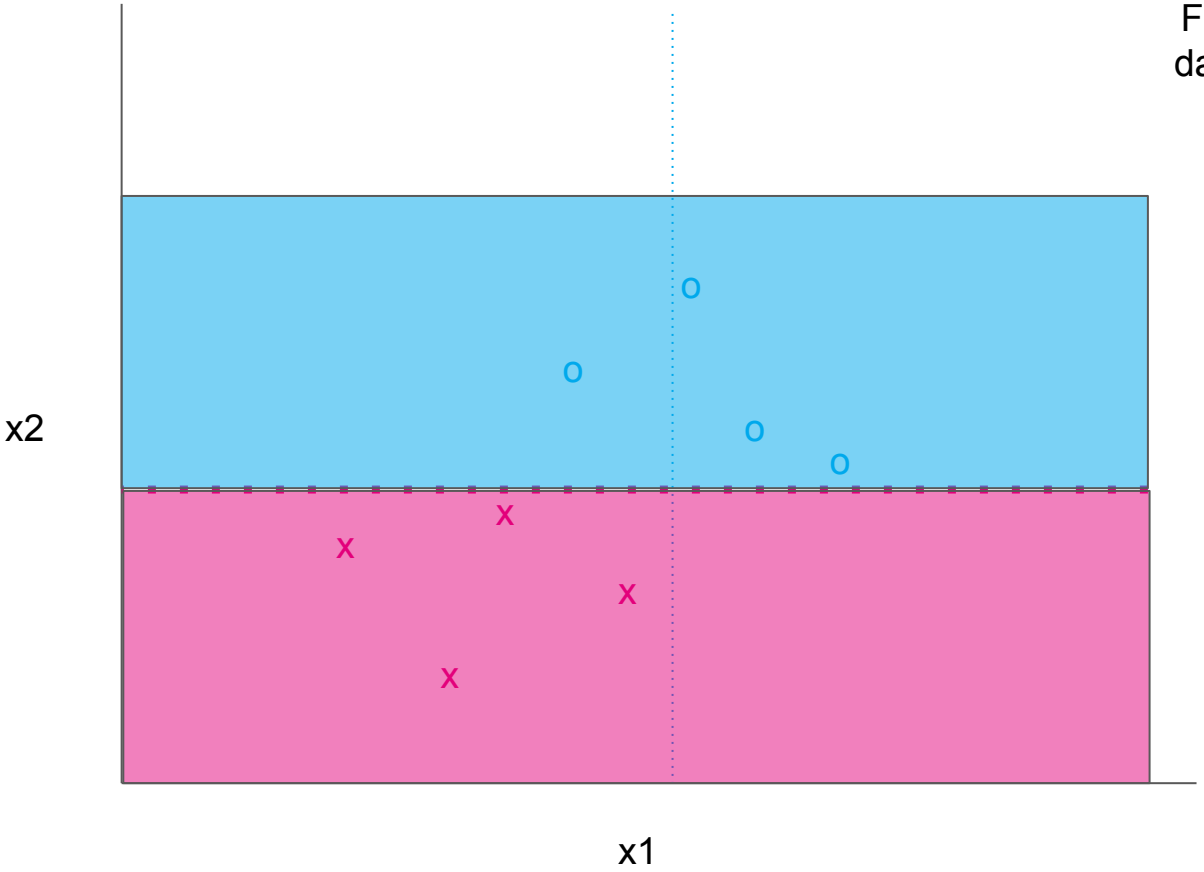

x2
x1
o
o
o
o
x
x
x
x


x2
x1
o
o
o
o
x
x
x
x

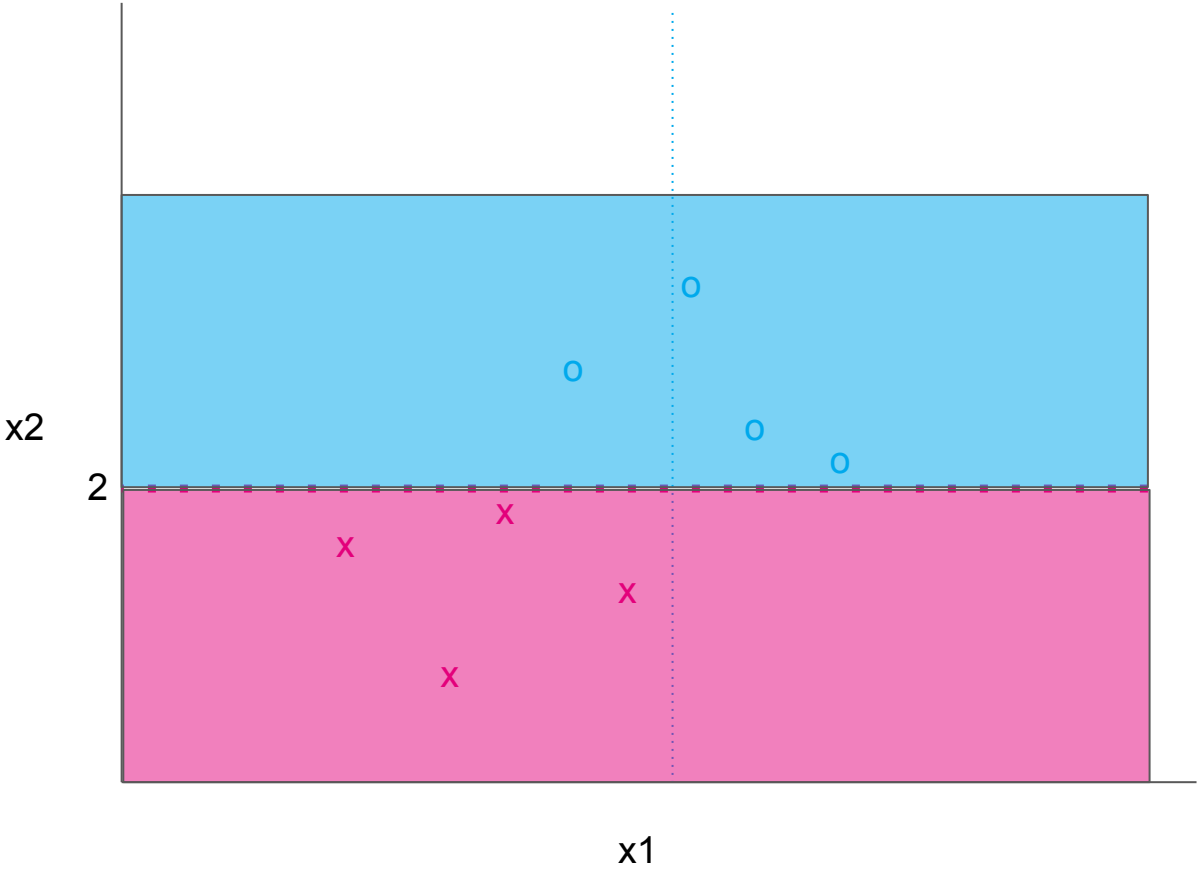

x2
x1
o
o
o
o
x
x
x
x

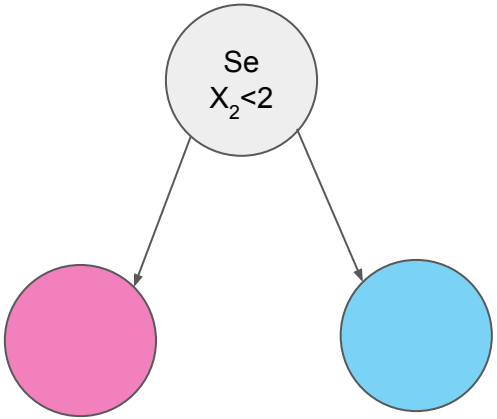

Melhor separação segundo
a feature x1
x2
x1
o
o
o
o
x
x
x
x


Agora, fazemos o
mesmo processo com a
feature x2
x2
x1
Melhor separação segundo
a feature x2



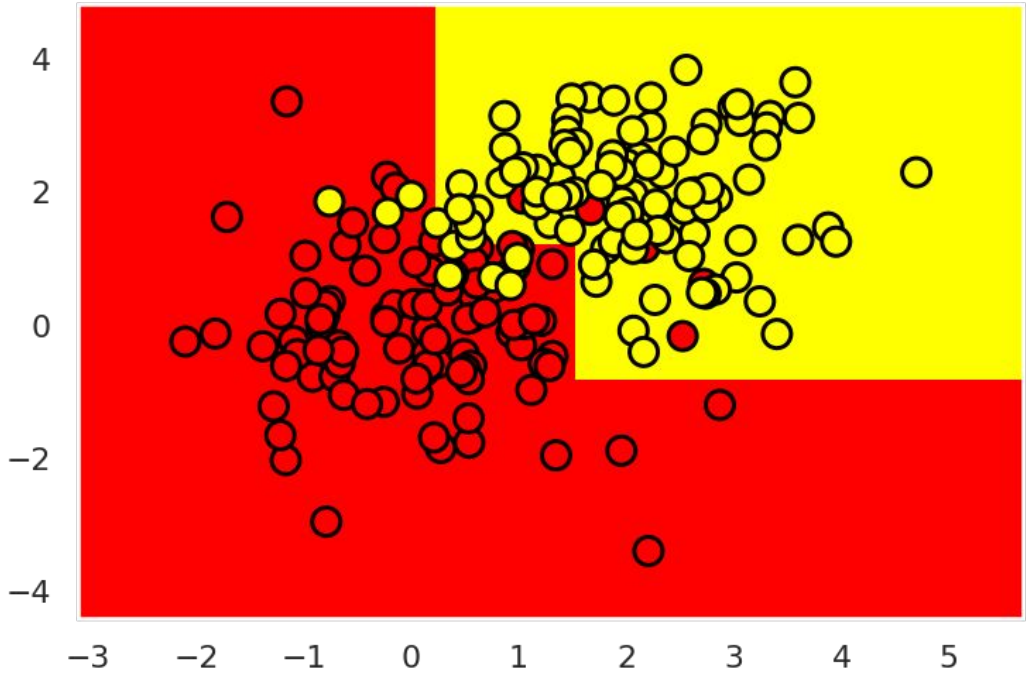
Feature x_2 separa melhor os dados, logo decidimos usá-la para dividir


x2
2
x1
o
o
o
o
x
x
x
x


Se
X2<2

Visualizando a superfície de decisão

Como os nós de decisão usam apenas comparações (< ou >) para cada feature isoladamente, a superfície de decisão vai ser construída por setores retângulares, com lados paralelos ao eixo.



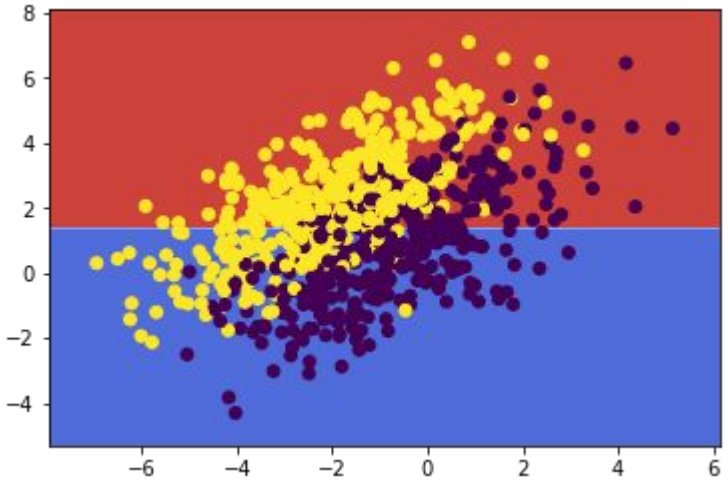
A capacidade de uma Árvore de Decisão

A capacidade de um modelo mede o quão complexas as suas predições podem ser. Em uma árvore de decisão, o principal método para controlarmos a capacidade do modelo é a altura máxima da árvore:

- Árvores com uma altura pequena terão poucos nós, então não serão capazes de aproximar funções mais complexas
- Árvores com uma altura grande podem ter muitos nós, permitindo mais complexidade

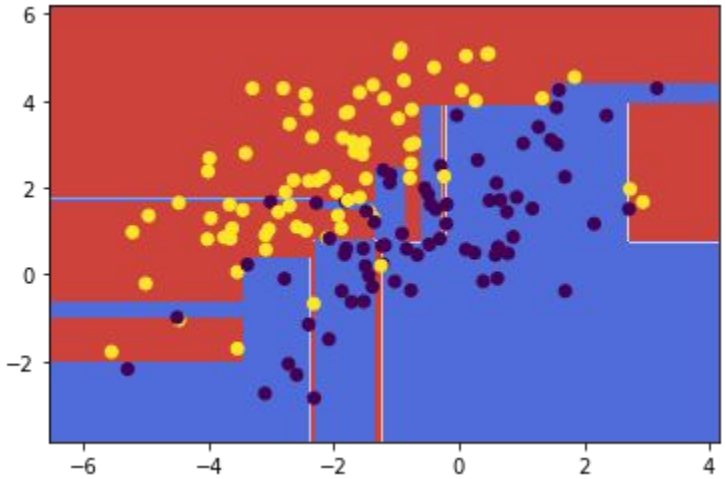
Exemplos

Uma árvore com um único nó tem uma capacidade extremamente pequena: ela só consegue separar a superfície uma única vez:



Exemplos

Uma árvore com muitos nós pode aprender superfícies extremamente complexas, chegando a uma acurácia de até 100% nos dados de treino:



Dilema Viés-Variância

As árvores de decisão tem um problema:

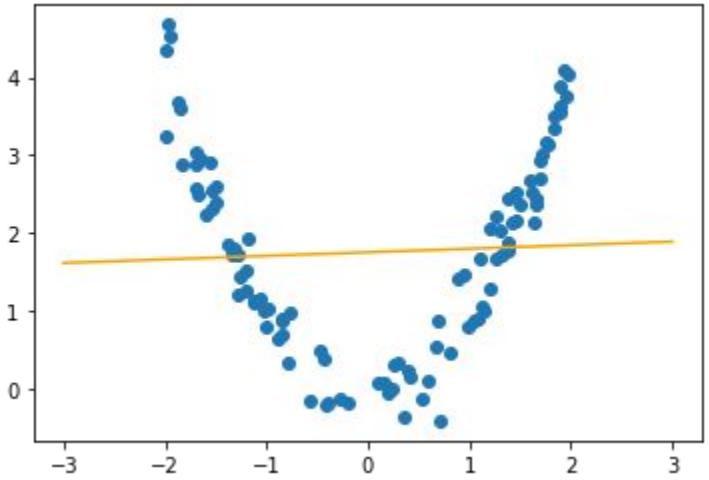
- Se usarmos uma altura muito pequena, a capacidade não será suficiente para conseguir uma performance boa
- Se usarmos uma altura muito grande, a capacidade será tão grande que o modelo irá memorizar os dados de treino, sem generalizar

Descrevendo os problemas

Existem duas fontes principais de problemas para os classificadores que levam a essas dificuldades: o viés e a variância.

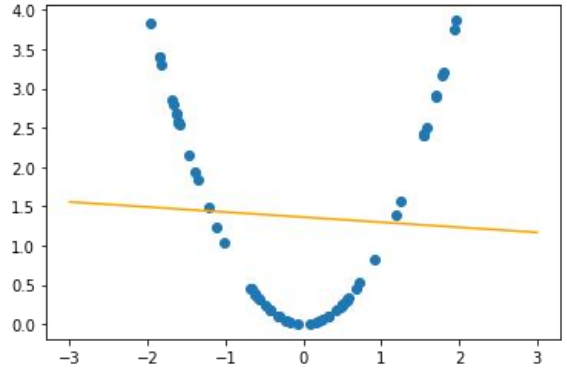
Classificadores Enviesados

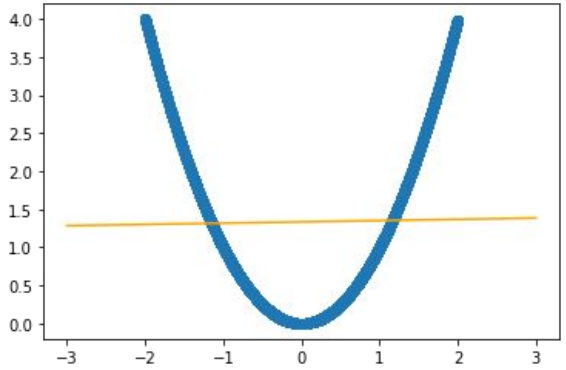
Classificadores enviesados são aqueles que não possuem capacidade suficiente para modelar a função verdadeira (a árvore de decisão com 1 nó é um exemplo).



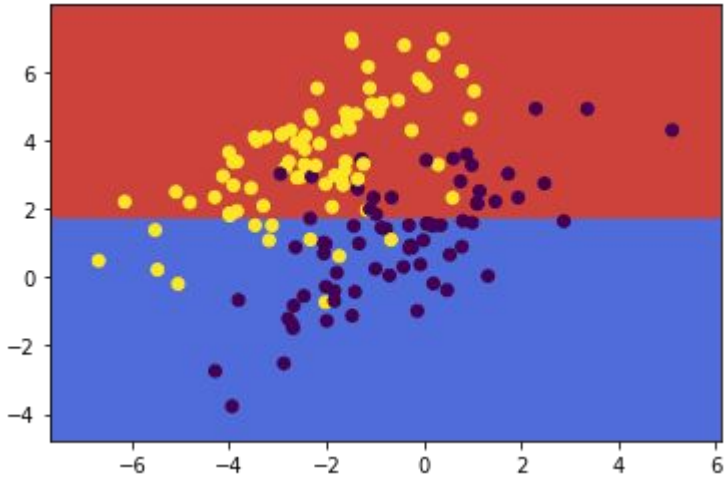
Desvantagens

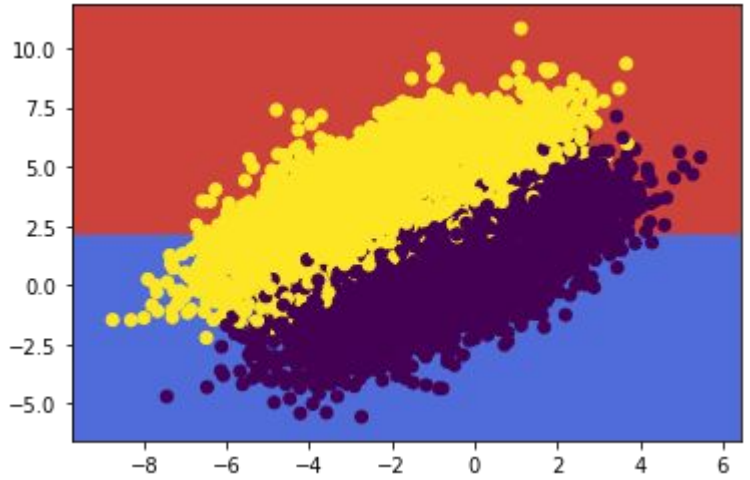
Por se tratar de uma limitação da própria capacidade do modelo, classificadores enviesados não conseguem uma acurácia alta mesmo quando a quantidade de dados aumenta:





Árvore de Decisão Enviesada





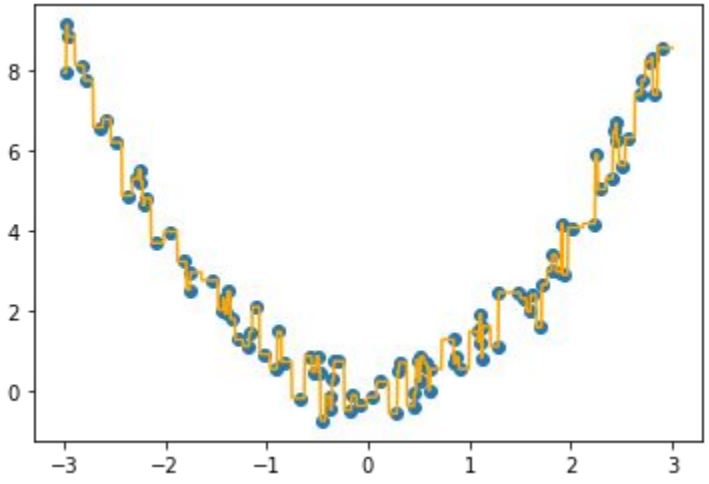
Underfitting

Apesar dos problemas, os classificadores enviesados têm a vantagem de não precisar de muitos dados para convergir pra uma solução, já que eles não sofrem com overfitting.

Apesar disso, eles sofrem do **underfitting**, já que não são capazes de aprender funções complexas.

Classificadores com alta Variância

Um classificador com alta variância consegue aprender funções altamente complexas, muitas vezes chegando até em um erro de treino 0.



Vantagens

Um classificador com alta variância que não seja enviesado (capacidade alta) pode encontrar uma solução com acurácia muito boa para o problema, desde que tenhamos dados suficientes.

Overfitting

O principal problema de um classificador com alta variância é o overfitting: o modelo vai incorporar padrões aleatórios do dataset que não representam a realidade, aumentando a acurácia no treino mas diminuindo a performance no teste.

A Variância

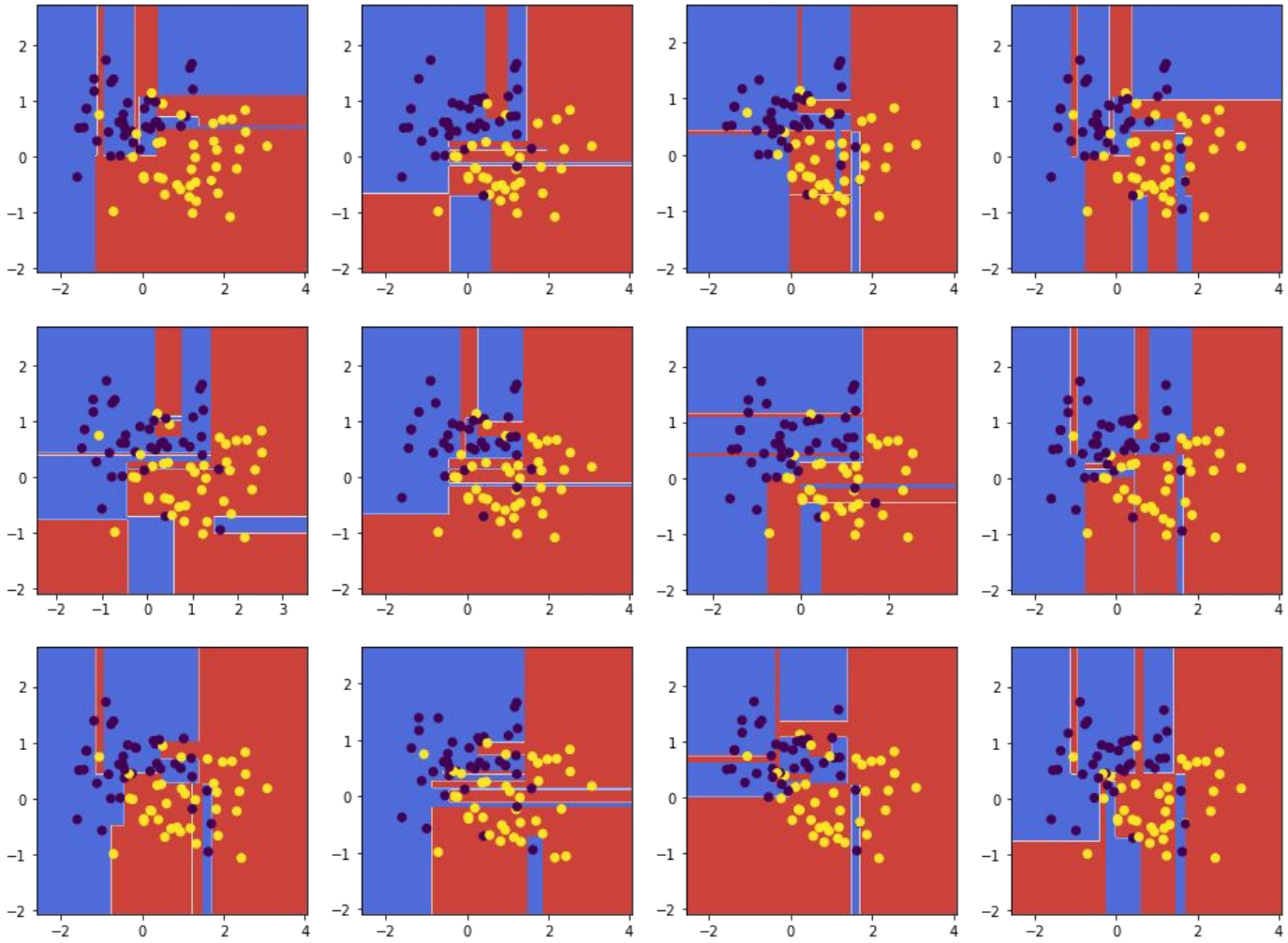
Nós podemos pensar no nosso dataset como uma **amostra** da distribuição dos dados verdadeira. Então fica claro o problema com classificadores de alta variância:

- Eles chegam a um erro de 0 na amostra, incorporando padrões aleatórios
- Por isso, as predições do modelo final variam muito dependendo do dataset de treino

Árvores de Decisão com Variância alta

As árvores de decisão com altura grande têm um problema com a variância alta, já que a árvore vai criar pequenos retângulos até que todos os pontos do treino sejam classificados corretamente e todas as regiões tenham 100% de pureza.

Porém, esses retângulos nos níveis mais baixos são muito específicos do dataset e quase com certeza não vão levar a uma performance melhor

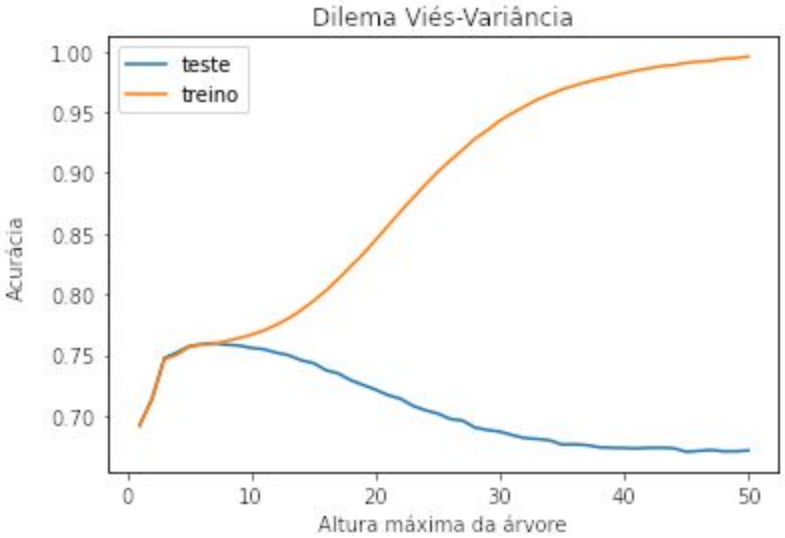


Dilema Viés-Variância

Árvores de decisão não são muito boas por si só, mas elas têm uma característica muito útil: podemos controlar diretamente a capacidade do modelo com o valor da altura máxima.

Porém, isso leva a um dilema: se diminuirmos muito o viés do modelo, isso irá aumentar a variância e vice-versa.

Dilema Viés-Variância



Como resolver o dilema

Tradicionalmente, nós tentamos encontrar a altura “perfeita” que equilibra o viés e a variância com um conjunto de validação, mas isso pode não funcionar tão bem. Nós temos outras possibilidades:

- Usar árvores de altura pequena (pouca capacidade, normalmente ocorre underfitting) e usar métodos que diminuem o viés.
- Usar árvores de altura grande (muita capacidade, normalmente ocorre overfitting) e usar métodos que diminuem a variância.

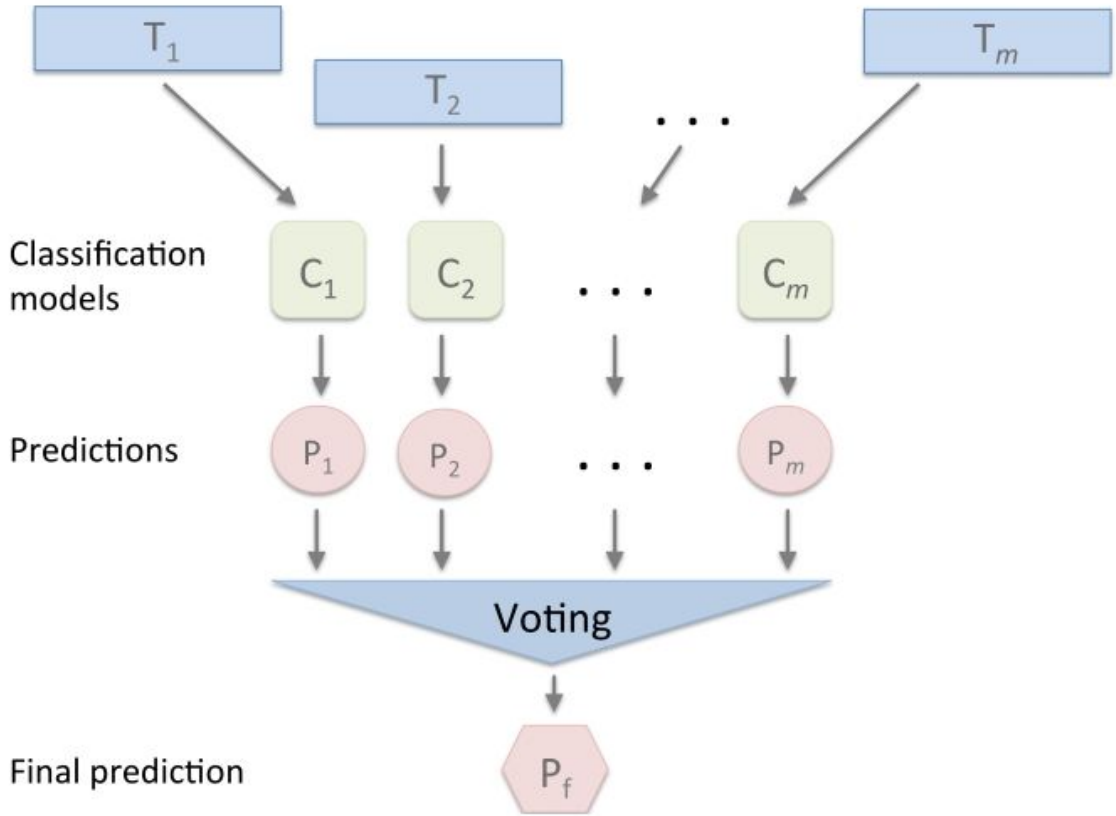
Reduzindo a Variância

Analisando o problema

A principal fonte do problema da alta variância é que os modelos ficam muito adaptados ao dataset usado no treino. Porém, seria possível reduzir essa variância se nós tivéssemos acesso a **mais de um dataset**.

Agregando Modelos

Suponha que nós temos acesso a N datasets de uma mesma tarefa. Nós podemos reduzir o problema da variância treinando um modelo em cada dataset separadamente, e no final calculando a **média** de todos os modelos treinados para a predição final.

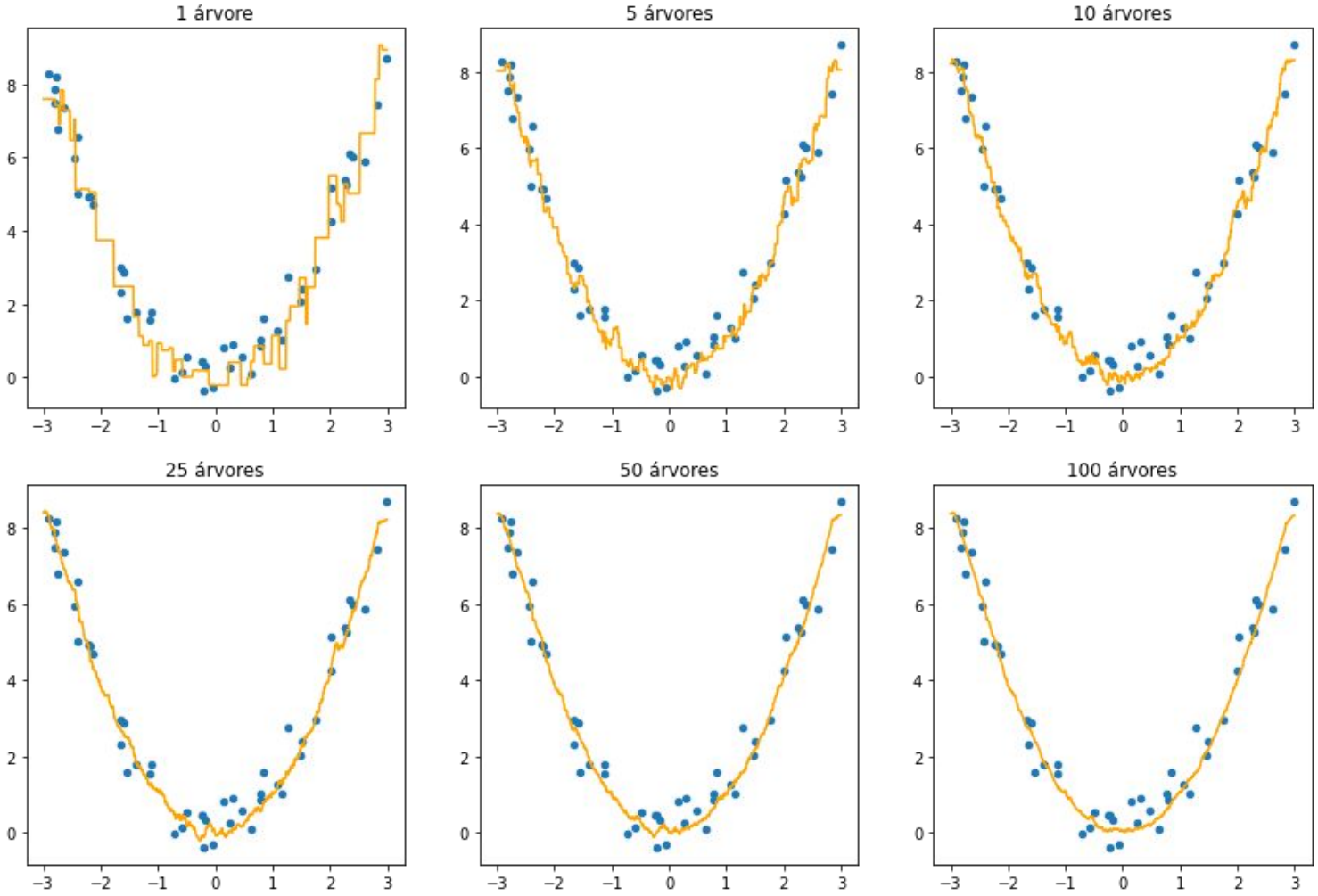

T1
T2
Tm
Classification models
C1
C2
Cm
Predictions
P1
P2
Pm
Voting
Final prediction
Pf

Explicando o funcionamento

Ao calcular a média, os erros dos modelos que são devido ao overfitting aos datasets cancelam uns aos outros, já que cada modelo é treinado em um conjunto diferente.

Porém, as predições que são sempre verdadeiras em todos os datasets vão ser amplificadas, já que a maioria dos modelos treinados vão concordar na saída.

Por isso, esse método permite o uso de modelos com uma capacidade muito maior, permitindo aprender padrões mais complexos, sem recair tanto no overfitting.


1 árvore
5 árvores
10 árvores
25 árvores
50 árvores
100 árvores

Na prática

Como vamos conseguir vários datasets adicionais?

Quebrar o nosso conjunto em partes não é uma opção, já que cada fração vai ficar muito pequena, fazendo o problema do overfitting retornar.

Também não podemos usar cópias idênticas do dataset, já que nós precisamos de diversidade nos conjuntos para que os erros devido ao overfitting se cancelem.

Bagging

Bootstrapping

O bootstrapping é um método da estatística que nos permite, a partir de um único dataset, obter vários outros conjuntos derivados do mesmo tamanho do original e com a mesma distribuição dos dados. Isso é possível porque os conjuntos derivados poderão ter elementos repetidos.

Os conjuntos derivados serão obtidos a partir da **amostragem** de elementos aleatórios do conjunto original.

Amostragem sem Reposição

Para os nossos propósitos, não é possível criar conjuntos derivados com amostragem sem reposição:

Amostragem sem Reposição

Amostragem sem Reposição

Amostragem sem Reposição

Amostragem sem Reposição

Amostragem sem Reposição

Problemas

O dataset final é idêntico ao original! Por isso, esse método não vai ser útil se quisermos gerar datasets diferentes para treinar os modelos.

Em vez disso, precisamos de **amostragem com reposição**, onde nós não eliminamos os itens já retirados do processo aleatório.

Amostragem com Reposição

Amostragem com Reposição

Amostragem com Reposição

Amostragem com Reposição

Amostragem com Reposição

Amostragem com Reposição

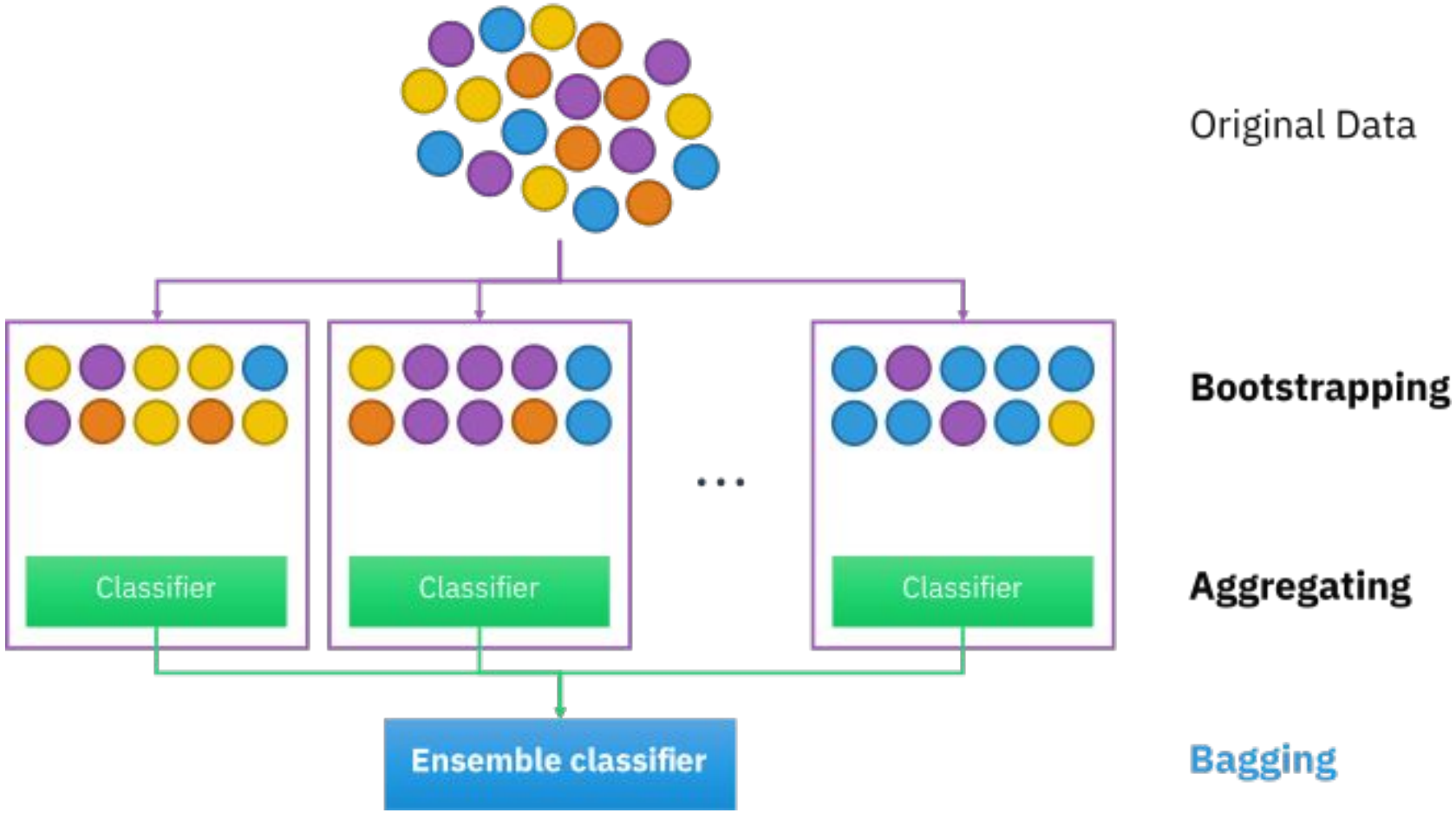
Bootstrapping

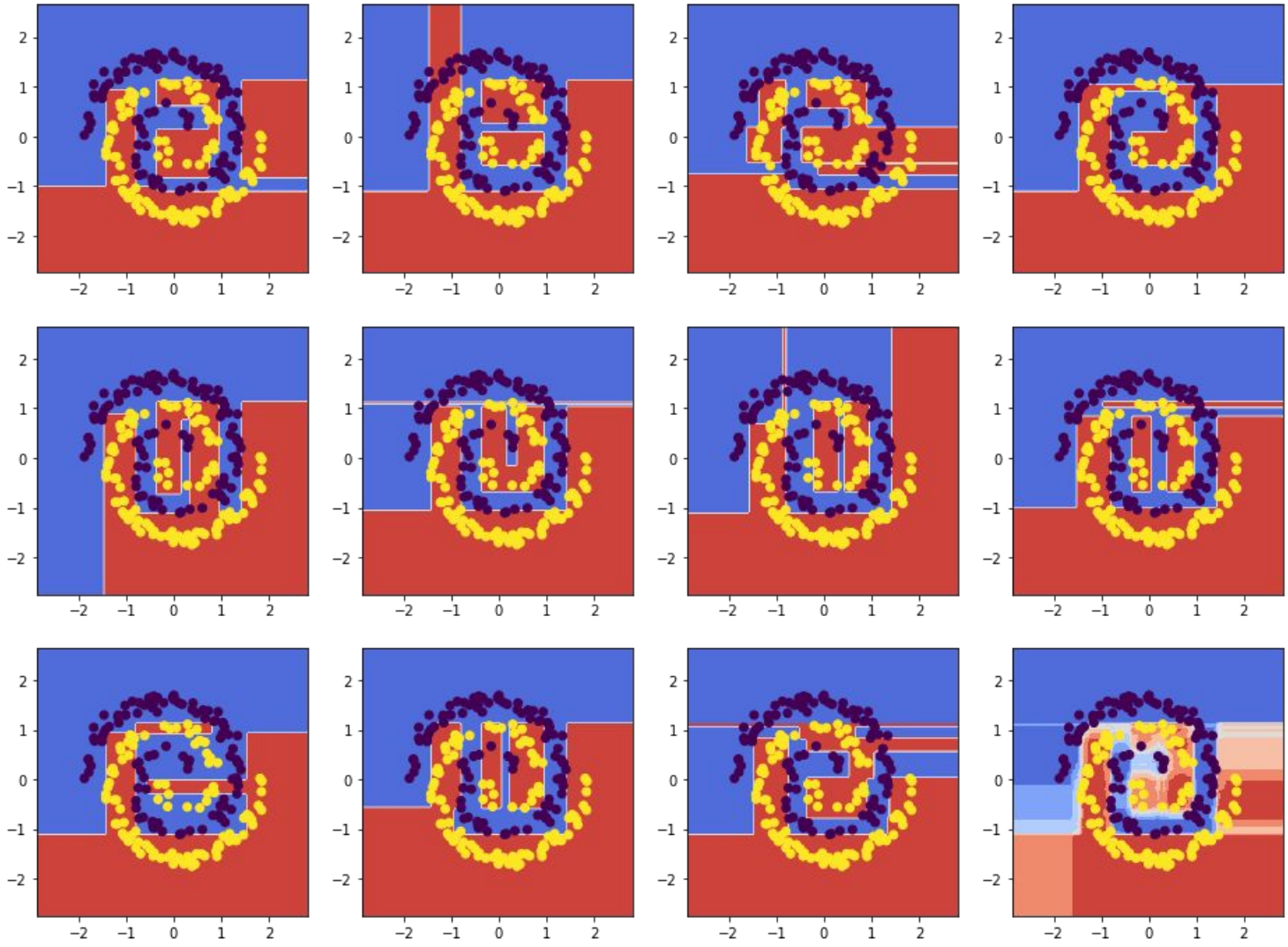
A cada vez que fizermos o processo de amostragem com reposição, o resultado final será diferente, o que é exatamente o que precisamos! Isso só é possível porque nós permitimos a existência de itens duplicados.

Em média, cada conjunto derivado terá **63.2%** dos itens do original, os restantes sendo duplicatas.

Bagging

Nós podemos unir o processo de Bootstrapping com a Agregação de modelos que vimos antes, para gerar o método Bagging (Bootstrap Aggregating), onde nós usamos o processo de amostragem com reposição para criar novos datasets nos quais treinamos os modelos.


Original Data
Bootstrapping
Classifier
Classifier
Classifier
Aggregating
Ensemble classifier
Bagging



Random Forests

A Diversidade da Ensemble

Para que as árvores na ensemble consigam compensar os erros umas das outras, é essencial que exista **diversidade** nos modelos. Isso porque se todas elas cometerem os mesmos erros, eles irão se acumular e o Bagging final também irá fazer a predição errada.

Por isso, nós queremos que as árvores sejam o mais independentes possíveis umas das outras, mas até agora a única fonte de diversidade são os diferentes conjuntos gerados por amostragem.

Uma fonte de correlações

Existe uma forte correlação entre os modelos treinados (o que diminui a diversidade) por um motivo: todas as árvores têm acesso a exatamente as mesmas features. Por isso, se uma feature for muito preditiva no treino, todas as árvores irão usá-la, diminuindo a diversidade.

Random Forests

O algoritmo Random Forests (florestas aleatórias) é uma variação do Bagging que aumenta a diversidade dos modelos treinados, sendo extremamente usado em ciência de dados.

Para isso, cada árvore na ensemble não terá acesso a todas as features. Em vez disso, nós selecionaremos, para cada uma, k features aleatórias, que serão as únicas visíveis para o modelo.
O valor de k é um hiperparâmetro, mas sqrt(N) é muitas vezes um bom valor inicial.

Erro Out-of-Bag

Medindo a performance

Normalmente, para avaliarmos um modelo, é necessário separar um conjunto de validação que não é usado durante o treino.

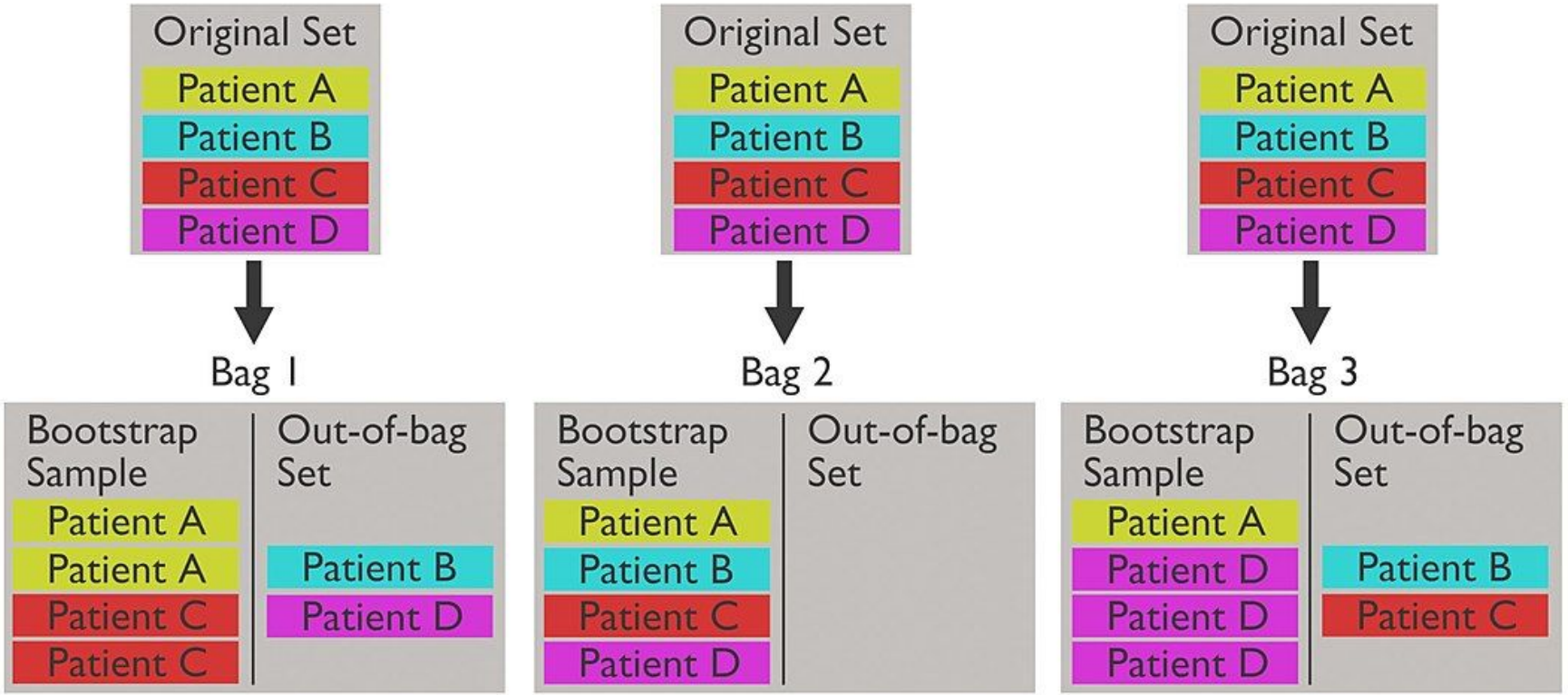
Porém, usando bagging, isso não é necessário, já que, como a amostragem é feita com reposição, as árvores individuais não tiveram acesso a todos os exemplos do dataset.

Erro Out-of-Bag

Nós podemos avaliar o funcionamento do modelo usando o erro Out-of-Bag.

Esse erro corresponde à métrica desejada (por exemplo, acurácia) calculada em cada ponto do dataset **apenas** com as árvores que não tiveram acesso a esse exemplo em sua bag no treinamento.

No final, basta fazer a média dos erros em todos os pontos.


Original Set
Patient A
Patient B
Patient C
Patient D
Bag 1
Bootstrap Sample
Patient A
Patient A
Patient C
Patient C
Out-of-bag Set
Patient B
Patient D
Original Set
Patient A
Patient B
Patient C
Patient D
Bag 2
Bootstrap Sample
Patient A
Patient B
Patient C
Patient D
Out-of-bag Set
Original Set
Patient A
Patient B
Patient C
Patient D
Bag 3
Bootstrap Sample
Patient A
Patient D
Patient D
Patient D
Out-of-bag Set
Patient B
Patient C

Porque isso funciona

- Um exemplo típico de sua classe, fácil de ser classificado
  - A maioria das árvores vai acertar a classe desse exemplo, levando a um bom erro Out-of-Bag.
- Um outlier, difícil de ser classificado
  - A maioria das árvores não vai conseguir acertar a classe desse exemplo, exceto as que viram ele no treinamento
- Um exemplo marcado com a label errada
  - Todas as árvores vão errar esse exemplo, exceto as que viram ele no treinamento

Observação

Com o erro Out-of-Bag, nós não precisamos do conjunto de validação, mas ainda precisamos do de teste, para evitar o overfitting com os hiperparâmetros.

Estimação de Incerteza

Incerteza

Algoritmos como Bagging e Random Forests têm ainda uma outra vantagem incrível sobre árvores individuais: elas oferecem uma estimativa da incerteza do modelo sobre a predição.

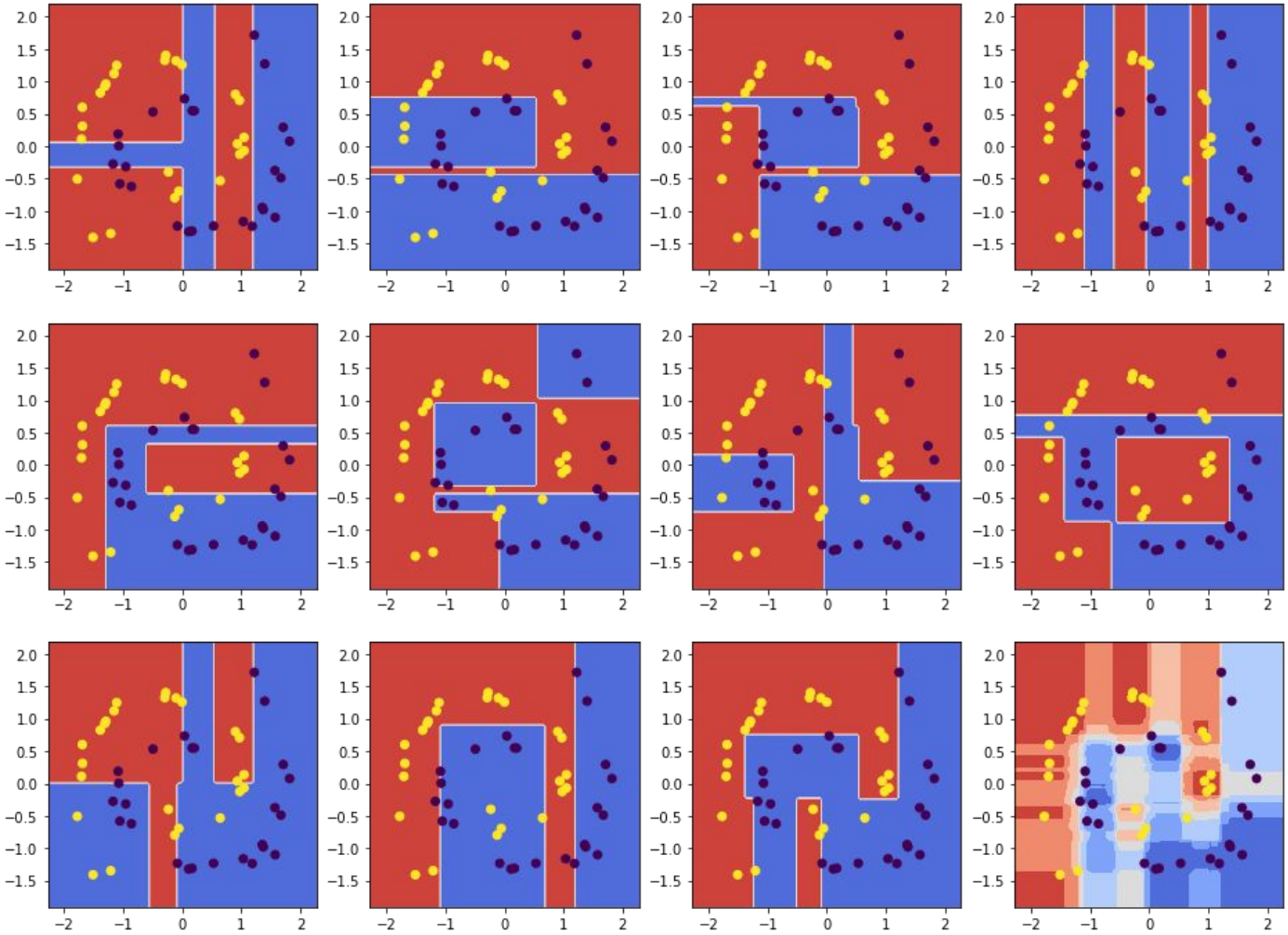
Isso porque uma árvore de decisão com muitos nós irá simplesmente oferecer como saída uma classe predita, mas para muitas aplicações é essencial sabermos o quão confiante o modelo está nessa predição.

Incerteza

Métodos de ensemble têm uma capacidade maior de estimar a incerteza:

- Se 100% das árvores fazem a mesma predição, ela tem grandes chances de ser verdadeira.
- Se 60% das árvores predizem a classe A e 40% a classe B, existe uma incerteza maior nessa predição,

Nesses casos, a incerteza da rede pode ser levada em consideração para uma eventual tomada de decisão, o que é essencial em aplicações sensíveis a erros.



Extra Trees

ExtraTrees

ExtraTrees é uma variação das Random Forests que aumenta extremamente a variabilidade dos modelos na ensemble, com uma alteração no processo de treinamento da árvore:

Para cada feature, em vez de testarmos cada posição do split possível para encontrar a que melhor divide os dados, simplesmente escolhemos uma aleatória! Nós ainda fazemos esse procedimento para cada feature e selecionamos a melhor delas.

ExtraTrees

Esse procedimento aleatório diminui a performance das árvores individuais, mas aumenta tanto a variabilidade da ensemble que não é necessário usar bagging: podemos sempre usar o conjunto todo.

Além disso, como não precisamos testar cada split, podemos treinar as árvores muito mais rapidamente, possibilitanto usar mais árvores na ensemble com o mesmo custo computacional durante o treino.