STT3030 - Cours #2

Arthur Charpentier

Automne 2024

Apprentissage supervisé (*Supervised learning*)

On veut prédire la réponse Y avec les prédicteurs $\mathbf{X}$:

- $\mathbf{X} = \{X_1, ..., X_p\} \in \mathcal{X}_1 \times ... \times \mathcal{X}_p = \mathcal{X}$ ($\mathcal{X}_j$ est le domaine de X_j) forment une collection de prédicteurs, variables explicatives, covariables, entrées, etc. (*predictors, features or input*),
- $Y \in \mathcal{Y}$ est une réponse, variable expliquée, étiquette (si catégorielle), ou sortie.

Pour cela, on suppose l'existence d'une fonction $f : \mathcal{X} \to \mathcal{Y}$ telle que $Y = f(\mathbf{X}) + \varepsilon$, avec ε une supposée variabilité inexpliquée (erreur de mesure, variabilité naturelle, etc.).

Exercice 2.4.2: Classification ou régression ?

1. We collect a set of data on the top 500 firms in the US. For each firm we record profit, number of employees, industry and the CEO salary. We are interested in understanding which factors affect CEO salary.
2. We are considering launching a new product and wish to know whether it will be a success or a failure. We collect data on 20 similar products that were previously launched. For each prod- uct we have recorded whether it was a success or failure, price charged for the product, marketing budget, competition price, and ten other variables.
3. We are interested in predicting the % change in the USD/Euro exchange rate in relation to the weekly changes in the world stock markets. Hence we collect weekly data for all of 2012. For each week we record the % change in the USD/Euro, the % change in the US market, the % change in the British market, and the % change in the German market.

$\rightarrow$ Ici, on s'intéresse à $\mathcal{Y}$.

Estimation et évaluation de f

1. Choix de la forme de f, du type de modèle: flexibilité, compromis biais/variance.
2. Estimation de f:
   - Données de validation (*validation set*): sélection des hyperparamètres du modèle (nombre de voisins pour kNN, etc.).
   - Données d'entraînement (*training set*): nous disposons d'un échantillon (*sample*) de données

     $s = \{s_i \mid i \in (1, \ldots, n)\} = \{(\mathbf{x}_i, y_i) \mid i \in (1, \ldots, n)\} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$,

     avec $s \in \mathcal{S} = \mathcal{X} \times \mathcal{Y} = \mathcal{X}_i \times \ldots \times \mathcal{X}_p \times \mathcal{Y}$, qui nous servira à entraîner notre modèle, c'est-à-dire apprendre à imiter f (attention au phénomène de surapprentissage).
3. Évaluation de f:
   - Données de test (*test set*): calcul de métriques telles que l'EQM (*MSE*) pour vérifier la généralisation du modèle.

Flexibilité: Exercice 2.4.1

2.4 Exercises

Conceptual

1. For each of parts (a) through (d), indicate whether we would generally expect the performance of a flexible statistical learning method to be better or worse than an inflexible method. Justify your answer.

   (a) The sample size n is extremely large, and the number of predictors p is small.

   (b) The number of predictors p is extremely large, and the number of observations n is small.

   (c) The relationship between the predictors and response is highly non-linear.

   (d) The variance of the error terms, i.e. $\sigma^2 = \text{Var}(\epsilon)$, is extremely high.

Figure 1: Extrait du livre ISLR

Grand n

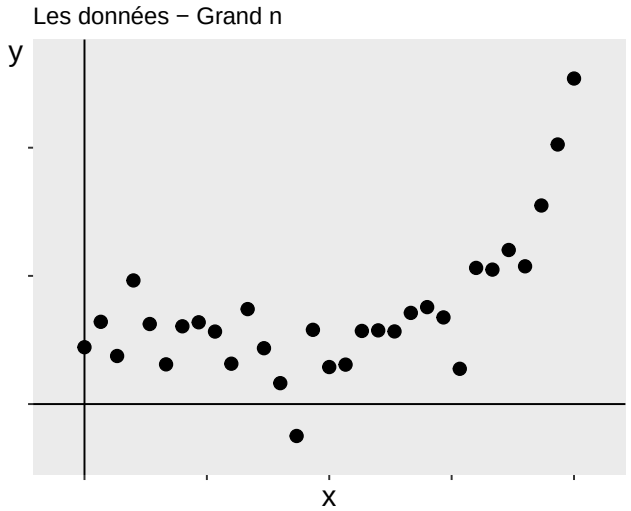


Figure 2: Données aves grand n

Petit n

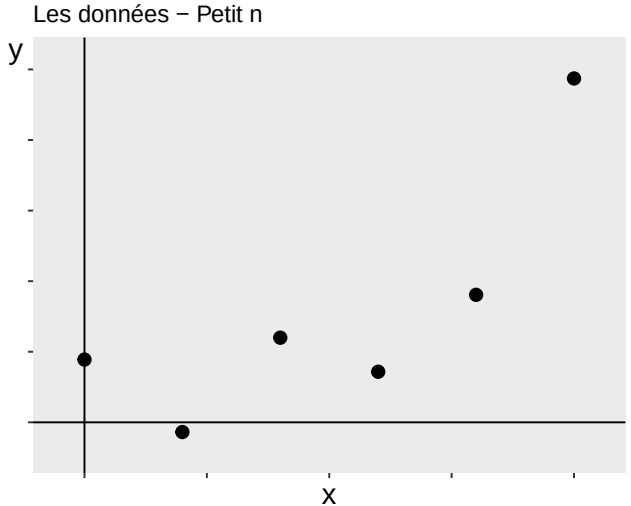


Figure 3: Données aves petit n

Petit n

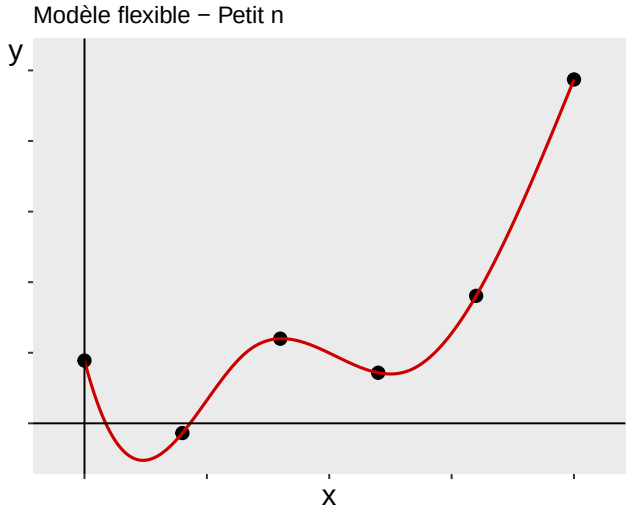


Figure 4: Données aves petit n

Petit n

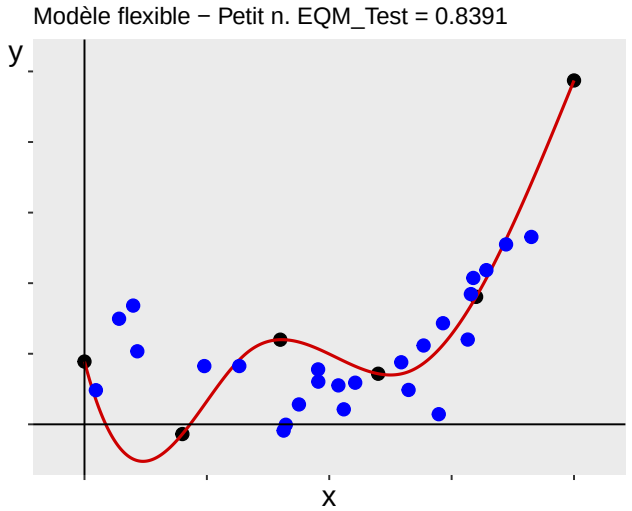


Figure 5: Données aves petit n

Petit n

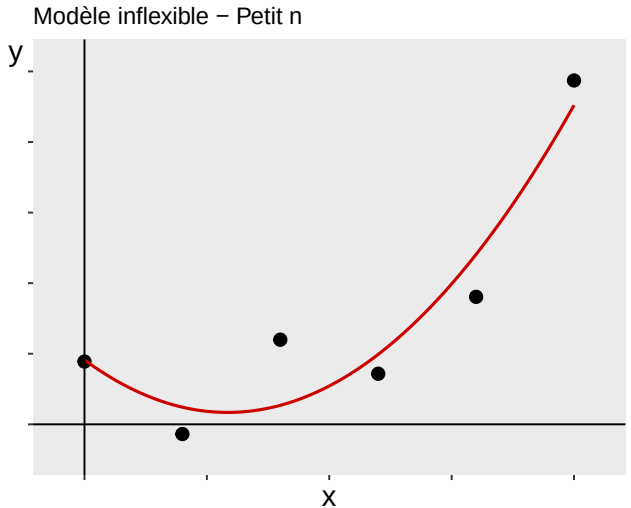


Figure 6: Données aves petit n

Petit n

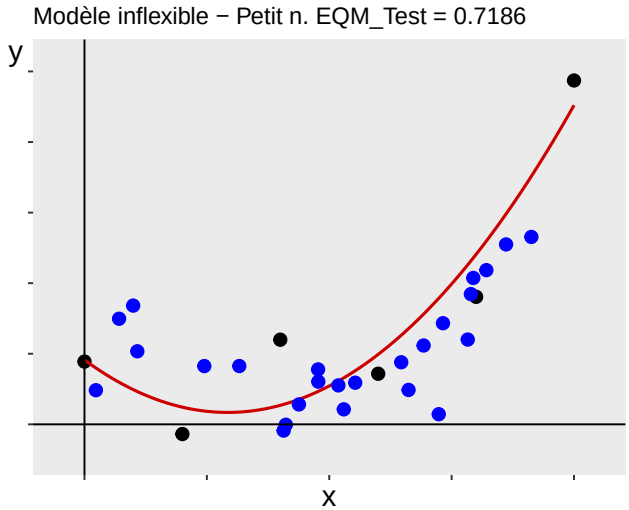


Figure 7: Données aves petit n

Grand n

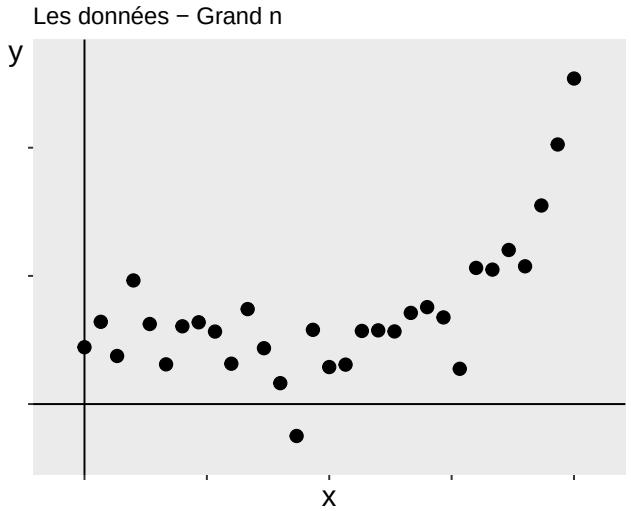


Figure 8: Données aves grand n

Grand n

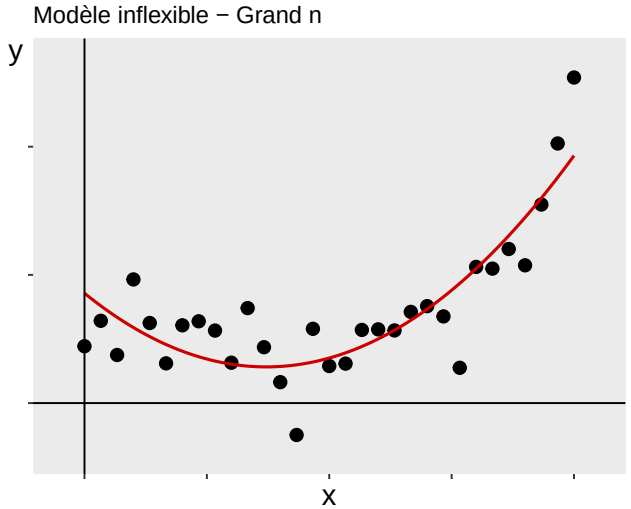


Figure 9: Données aves grand n

Grand n

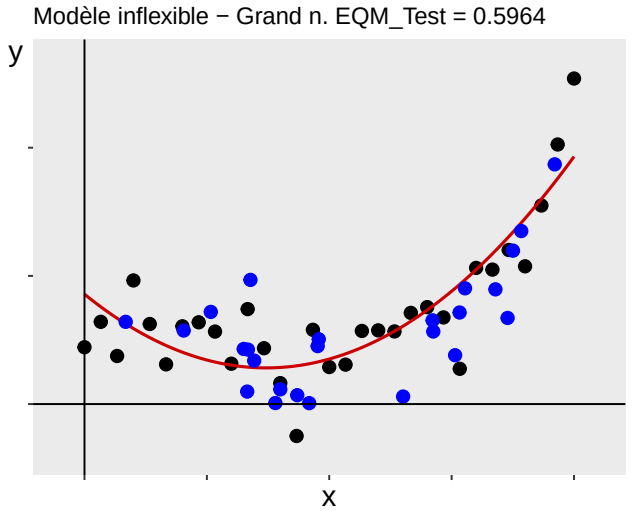


Figure 10: Données aves grand n

Grand n

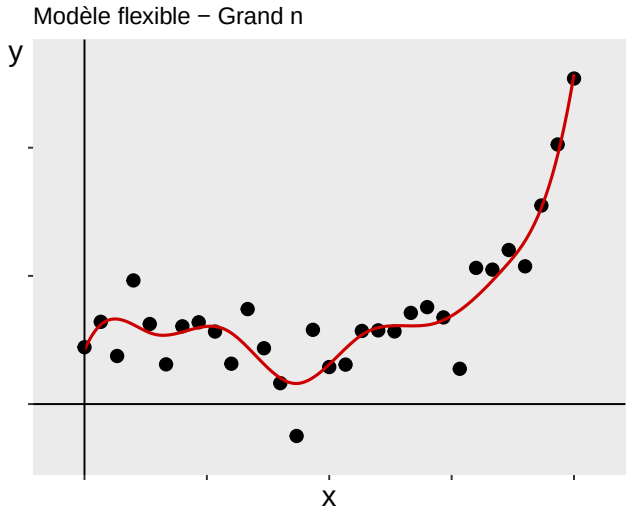


Figure 11: Données aves grand n

Grand n

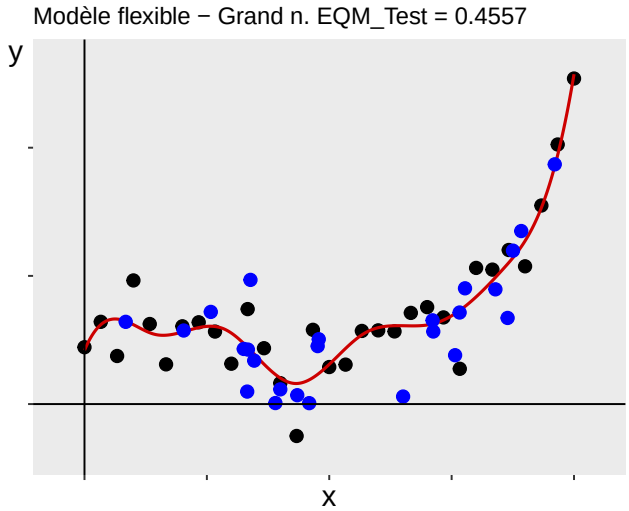


Figure 12: Données aves grand n

Grand ε

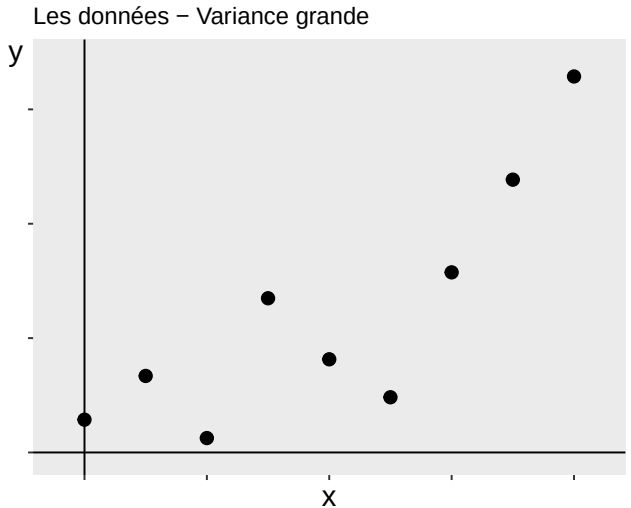


Figure 13: Données aves grand ε

Grand ε

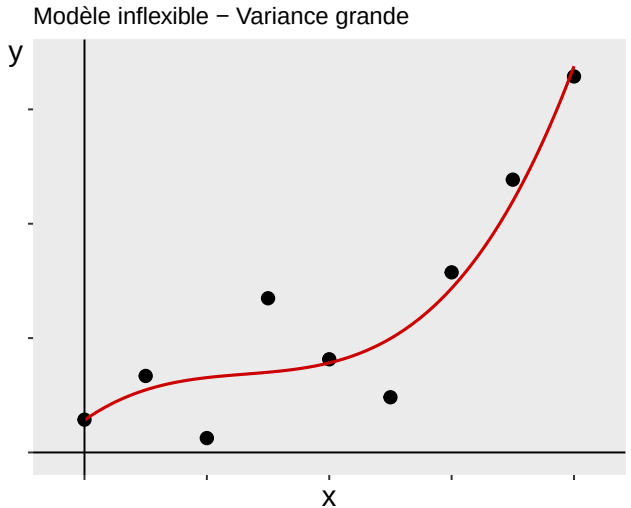


Figure 14: Donnée aves grand ε

Grand ε

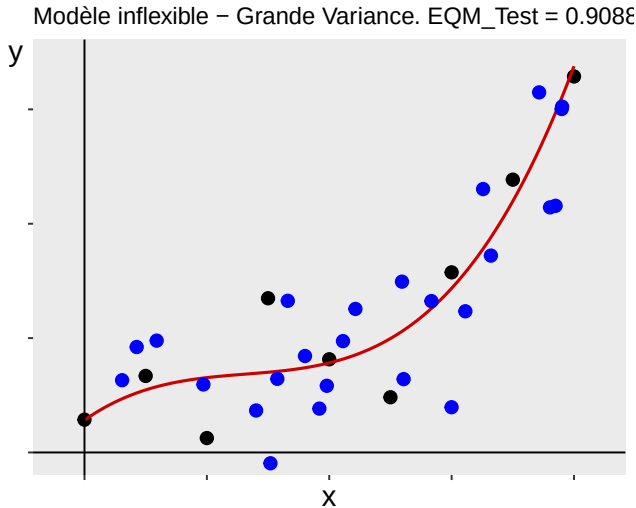


Figure 15: Donnée aves grand ε

Grand ε

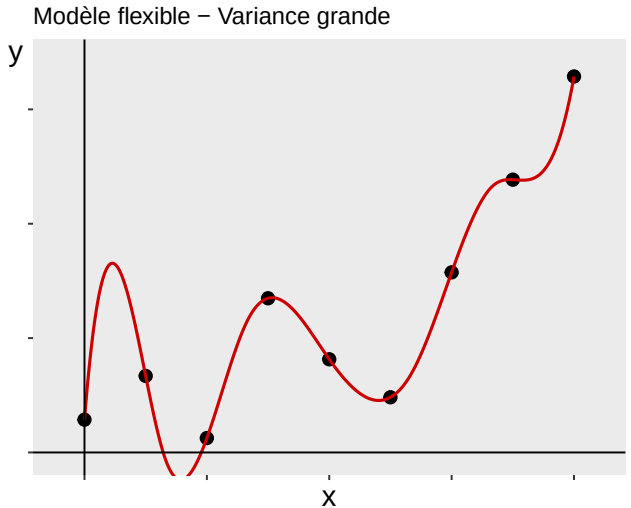


Figure 16: Donnée aves grand e

Grand ε

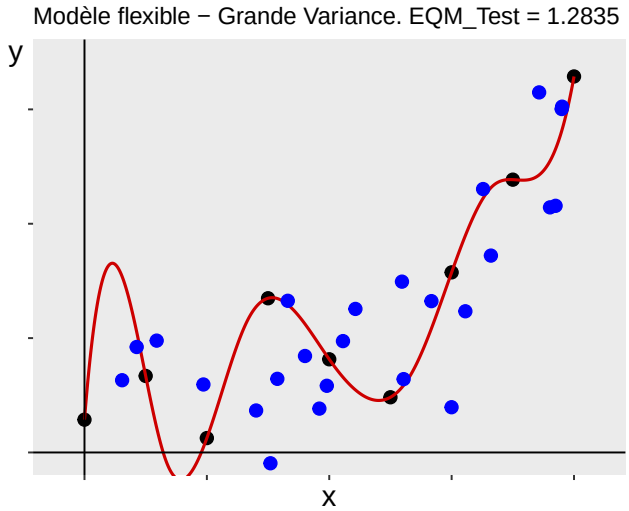


Figure 17: Donnée aves grand e

Modèle linéaire

Ici, on suppose $Y = \mathbf{X}^T\beta + \varepsilon$, où $\beta \in \mathbb{R}^{p+1}$. On sélectionne les paramètres β minimisant l'EQM sur les données d'entraînement. On cherche donc à résoudre le problème d'optimisation suivant:

$$\underset{\beta}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 = \underset{\beta}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n}(Y_i - \mathbf{X}_i^T\beta)^2$$

La droite des moindres carrés est la droite (ou le plan) retourné par ce problème d'optimisation:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T Y$$

- Cette droite EST TOUJOURS celle des moindres carrés peu importe la distribution de l'erreur.
- Une fomulation d'optimisation telle que la minimisation de l'EQM d'entraînement se porte bien à de la régularisation.

Modèle linéaire gaussien

Modèle linéaire: Ici, on suppose $Y = \mathbf{X}^T\beta + \varepsilon$, où $\beta \in \mathbb{R}^{p+1}$.

On suppose de plus $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. Par conséquent, $Y \sim \mathcal{N}(\mathbf{X}^T\beta, \sigma^2)$ et $\hat{\beta}$ s'obtient en maximisant la vraisemblance:

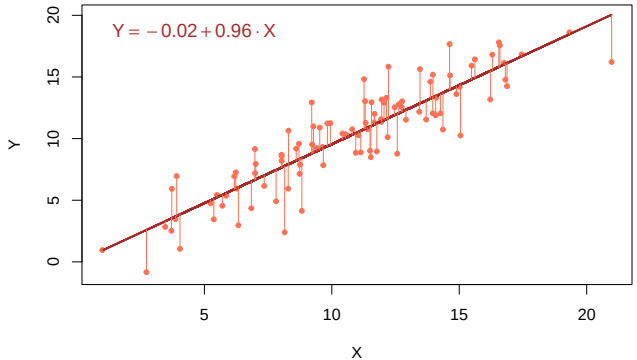
$$\hat{\beta} = \underset{\beta}{\text{argmax}} \prod_{i=1}^{n} p(y_i; \beta) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T Y$$

avec p fonction de densité de Y, suivant une loi normale.

$\rightarrow$ Avec cette hypothèse en plus, on retrouve la droite des moindres carrés. Mais cette fois-ci, on est en mesure de faire de l'inférence; puisqu'on connaît la distribution de $\hat{\beta}$.

En pratique

```
> fit <- lm(Y ~ X)
> b0 <- fit$coefficients[1]
> b1 <- fit$coefficients[2]
> Y_pred <- b0 + b1*X
> plot(X, Y, pch = 20, col = "coral1", ylim = c(-1,20))
> lines(X, Y_pred, col = "brown", lwd = 2)
```



Régression linéaire: La suite

- Manipulation des prédicteurs: prédicteurs catégoriels, intéraction, régression polynomiale,
- Manipulation des réponses possibles: régression logistique et régression généralisée.

Variable qualitative ou catégorielle binaire (1/3)

Supposons que nous voulons prédire le poids d'un animal Y et que nos variables explicatives sont les suivantes, sa taille X_1 et le type d'animal X_2 ("Chat" ou "Chien").

En régression linéaire, on prend une combinaison linéaire des variables explicatives et des coefficients :

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2$$

Comment prendre une combinaison linéaire de X_2 ?

Exemple: $y_1 = 25 + 2.3 \times 111 + 0.5 \times$ "Chat"

$\rightarrow$ On utilise une variable nominale *dummy variable*.

Variable qualitative ou catégorielle binaire (2/3)

On va prendre:

$$X_2 = \begin{cases} \text{Chat} \\ \text{Chien} \end{cases}$$

et on la transforme en:

$$X_2^* = 1_{\{X_2 = \text{Chien}\}} = \begin{cases} 0 \text{ si } X_2 = \text{Chat} \\ 1 \text{ si } X_2 = \text{Chien} \end{cases}$$

Nouveau modèle avec X_2^*:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2^*$$

Variable qualitative ou catégorielle binaire (3/3)

Par conséquent, nous aurons le modèle suivant:

$$Y = \begin{cases} \beta_0 + \beta_1 X_1 \text{ pour les Chats,} \\ \beta_0 + \beta_2 + \beta_1 X_1 \text{ pour les Chiens.} \end{cases}$$

$\rightarrow$ On fait simplement une translation de β_2.

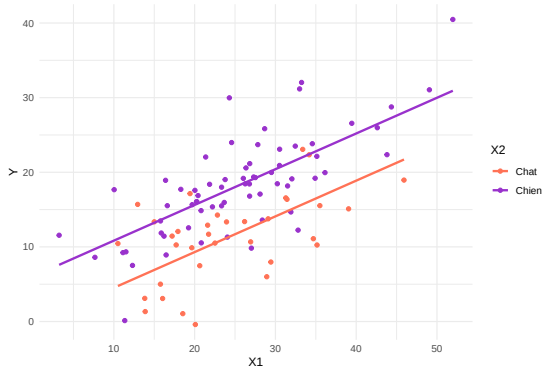
En pratique (1/2)

Sur R, il n'est pas utile de transformer nous-même X_2, il faut simplement préciser que c'est une variable catégorielle.

```
> X2 <- as.factor(X2)
> fit <- lm(Y ~ X1 + X2)
> summary(fit)
Call:
lm(formula = Y ~ ., data = data)
Residuals:
     Min       1Q   Median       3Q      Max
-11.3589  -2.4968   0.0824   2.0768  12.2888
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.26905    1.46444  -0.184    0.855
X1           0.47862    0.05074   9.432 2.26e-15 ***
X2Chien      6.32067    0.98417   6.422 4.98e-09 ***
---
...
```

En pratique (2/2)

```
> library(ggplot2)
> Y_pred <- fit$fitted.values
> df <- data.frame(Y, X1, X2, Y_pred)
> ggplot(df, aes(x = X1, y = Y, color = X2)) + geom_point() +
    geom_line(aes(y = Y_pred), size = 1) + scale_color_manual(values =
     c("Chat" = "coral1", "Chien" = "darkorchid"))
```



Variable catégorielle: Cas général (1/4)

Supposons que X_2 contient $k = 3$ catégories ("Chat", "Chien", "Lapin") et $n = 4$ individus.

Initialement, la matrice des covariables est:

$$\mathbf{X} = (X_1, X_2) = \begin{pmatrix} 20 & \text{"Chien"} \\ 50 & \text{"Chat"} \\ 25 & \text{"Chat"} \\ 35 & \text{"Lapin"} \end{pmatrix}$$

Variable catégorielle: Cas général (2/4)

La méthode la plus naturelle est de découper X_2 en 3 variables $X_{\text{Chat}} = 1_{\{X_2=\text{Chat}\}}$, $X_{\text{Chien}} = 1_{\{X_2=\text{Chien}\}}$ et $X_{\text{Lapin}} = 1_{\{X_2=\text{Lapin}\}}$.

Ainsi, nous obtenons une matrice $\mathbf{X}$ de taille $n \times (1+k)$, i.e. 4×4:

$$\mathbf{X} = (X_1, X_{\text{Chat}}, X_{\text{Chien}}, X_{\text{Lapin}}) = \begin{pmatrix} 20 & 0 & 1 & 0 \\ 50 & 1 & 0 & 0 \\ 25 & 1 & 0 & 0 \\ 35 & 0 & 0 & 1 \end{pmatrix}$$

Modèle: $Y = \beta_0 + \beta_1 X_1 + \sum_{j \in \{\text{Chat, Chien, Lapin}\} \text{ de taille } k} \alpha_j X_j$.

Attention à la colinéarité de $\mathbf{X}$: $X_{\text{Chat}} = 1 - X_{\text{Chien}} - X_{\text{Lapin}}$, on ne peut pas inverser la matrice $\mathbf{X}^T\mathbf{X}$. Ça ne posera pas de problèmes pour les arbres de décision et les réseaux de neurones.

Variable catégorielle: Cas général (3/4)

Finalement, découper X_2 en 2 variables binaires $X_{\text{Chien}} = 1_{\{X_2=\text{Chien}\}}$ et $X_{\text{Lapin}} = 1_{\{X_2=\text{Lapin}\}}$ ne nous fait perdre aucune information et évite le problème de colinéarité.

Ainsi, nous obtenons une matrice $\mathbf{X}$ de taille $n \times (1 + (k-1))$, i.e. 4×3:

$$\mathbf{X} = (X_1, X_{\text{Chien}}, X_{\text{Lapin}}) = \begin{pmatrix} 20 & 1 & 0 \\ 50 & 1 & 0 \\ 25 & 0 & 0 \\ 35 & 0 & 1 \end{pmatrix}$$

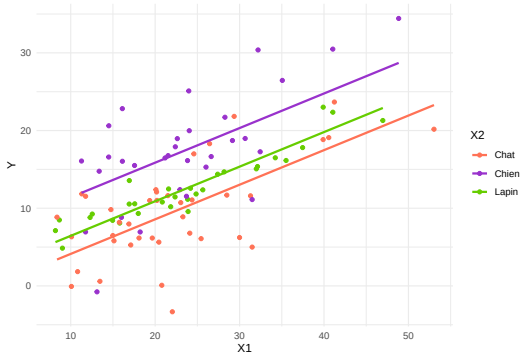
Modèle: $Y = \tilde{\beta}_0 + \beta_1 X_1 + \sum_{j \in \{\text{Chien, Lapin}\} \text{ de taille } k-1} \tilde{\alpha}_j X_j$.

"Chat" est appelée modalité de référence et son influence est contenue dans $\tilde{\beta}_0$.

Variable catégorielle: Cas général (4/4)

Maintenant, nous pouvons apprendre un modèle linéaire de la forme:

$$Y = \begin{cases} \beta_0 + \beta_1 X_1 \text{ pour les Chats,} \\ \beta_0 + \alpha_1 + \beta_1 X_1 \text{ pour les Chiens,} \\ \beta_0 + \alpha_2 + \beta_1 X_1 \text{ pour les Lapins.} \end{cases}$$



En pratique

En pratique, comme dans le cas binaire, on n'a généralement pas besoin de créer manuellement ce vecteur.

```
> X2 <- as.factor(X2)
> fit <- lm(Y ~ X1 + X2)
> summary(fit)
Call:
lm(formula = Y ~ ., data = data)
Residuals:
     Min       1Q   Median       3Q      Max
-13.5623  -1.7427   0.1122   2.3205   9.0849
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.33231    1.25588  -0.265    0.792
X1           0.44525    0.04667   9.540 1.45e-15 ***
X2Chien      7.29249    1.05821   6.891 5.79e-10 ***
X2Lapin      2.32512    1.06870   2.176    0.032 *
---
...
```

Intéraction (1/2)

Supposons qu'on aimerait capturer l'effet qu'a un prédicteur, disons X_2, sur la relation entre un autre prédicteur, disons X_1, et Y.

On peut faire ça en restant dans la famille des modèles linéaires en introduisant un terme d'intéraction linéaire dans l'équation: $X_1 \times X_2$.

Soit le **modèle de base**:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \quad .$$

On veut permettre à β_1 de varier en fonction de X_2.

Soit le **modèle tenant compte de cette intéraction**:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \gamma X_1 X_2 \quad .$$

Intéraction (2/2)

On peut facilement apprendre γ en multipliant X_1 et X_2 terme à terme; $X^* = X_1 \times X_2$, puis en faisons une régression linéaire:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \gamma X^* \quad .$$

Cette intéraction reflète que l'effet de X_1 sur Y varie selon la valeur de X_2 et vice versa.

Exemple

Prenons l'exemple $Y = \text{Salaire}$, $X_1 = \text{Heures de travail}$ (variable quantitative) et $X_2 = \text{Sexe} \in \{\text{Homme, Femme}\}$ (variable qualitative).

Sans tenir compte de l'intéraction:

- Les employés travaillant plus d'heures par semaine peuvent s'attendre à un salaire plus élevé.
- Les hommes et les femmes peuvent avoir des niveaux de salaire différents en raison de divers facteurs comme les écarts salariaux entre les sexes.

Pourquoi l'intéraction est importante ?
Les femmes pourraient avoir une relation différente des hommes entre les heures travaillées et le salaire, possiblement en raison de différences dans les types d'emplois ou les opportunités de travail supplémentaires.

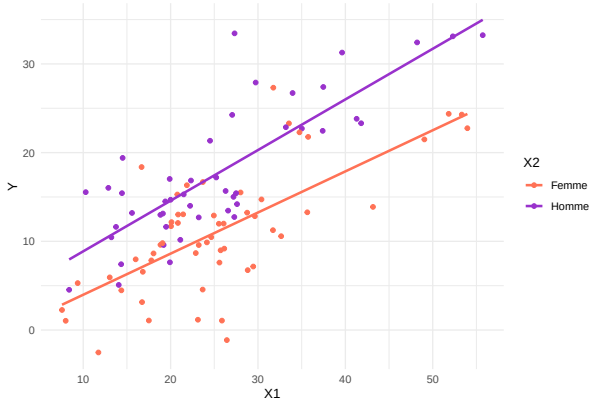
En pratique (1/3)

- En pratique on n'a pas besoin de faire le produit des variables, R/Python s'occupe de gérer l'intéraction, mais on doit le lui demander.

```
> fit <- lm(Y ~ X1*X2)
> summary(fit)
Call:
lm(formula = Y ~ X1 * X2, data = data)
Residuals:
     Min       1Q   Median       3Q      Max
-12.5441  -2.7160   0.1142   2.9111  11.6859
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.11076    1.21099   1.743   0.0845 .
X1           0.40283    0.04632   8.696 9.37e-14 ***
X2Homme      1.61797    5.42084   0.298   0.7660
X1:X2Homme   0.22910    0.18110   1.265   0.2089
---
...
```

En pratique (2/3)

- En ajoutant une intéraction, contrairement à l'exemple précédent, on autorise la pente (sur X_1) à varier suivant la modalité de X_2 (Homme/Femme).

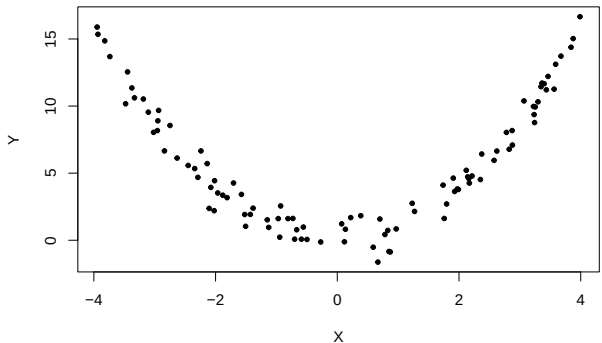


En pratique (3/3)

- Les termes d'intéraction existent entre plusieurs variables quantitatives ou plusieurs variables catégorielles, et également entre variables quantitatives et variables catégorielles.
- Les intéractions peuvent être aussi de troisième degré ($X_1 \times X_2 \times X_3$) et encore plus. En ajoutant les termes d'intéraction, on augmente la complexité/flexibilité du modèle linéaire puisqu'on ajoute des paramètres à apprendre. On doit donc déterminer quelles intéractions inclure pour ne pas rendre le modèle trop difficile à interpréter.

Aller au-delà de la linéarité : Régression polynomiale

Chapitre 7.1 du livre James et al. (2013).



→ La relation entre Y et X semble être non linéaire.

Régression polynomiale

On peut apprendre une fonction linéaire entre Y et $\mathbf{X}^d = (X_1^d, X_2^d, \cdots, X_p^d)$ pour quelconque puissance $d \geq 1$.

Par exemple, on prend une variable (numérique) X_j, $j \in \{1, \cdots, p\}$, on augmente sa puissance en calculant X_j^d pour chaque observation.

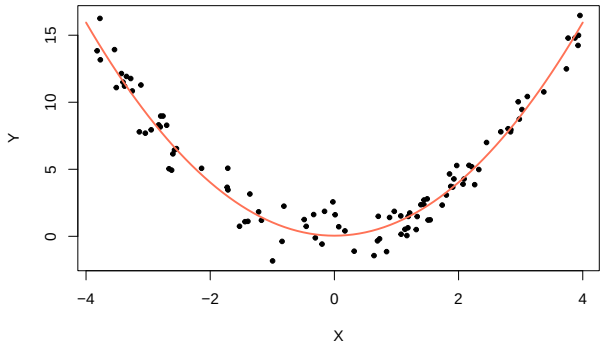
Puis, on apprend un modèle de régression linéaire entre Y et X_j^d:

$$Y = \beta_0 + \beta_1 X_j^d$$

En pratique

Ici, $d = 2$.

```
> X2 <- X^2
> fit <- lm(Y~X2)
```



Régression polynomiale

On peut également apprendre le modèle composé de toutes les puissances de X_j jusqu'à d:

$$Y = \beta_0 + \beta_1 X_j + \beta_2 X_j^2 + \beta_3 X_j^3 + \cdots + \beta_d X_j^d$$

On inclut donc X_j^1, X_j^2, X_j^3, jusqu'à X_j^d comme variables explicatives de Y.

Comment chosir d, le degré du polynôme que nous incluons dans le modèle ?

Régression polynomiale

Prenons un jeu de données contenant $n = 10$ observations, sur lequel on apprend un modèle de régression polynomiale à partir de X pour prédire Y.

Faisons varier le degré du polynôme: $d = 2$, $d = 3$ et $d = 10$.

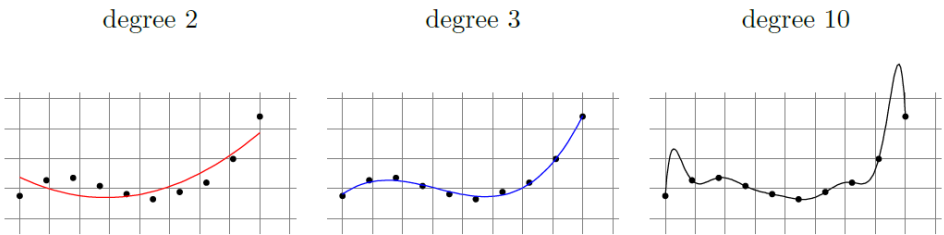


Figure 18: Extrait du livre Shalev-Shwartz and Ben-David (2014)

Régression polynomiale

Les différentes courbes de régression obtenues sur les données d'entraînement sont les suivantes:

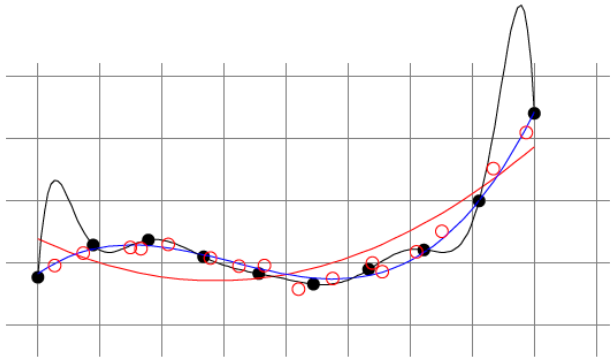


Figure 19: Extrait du livre Shalev-Shwartz and Ben-David (2014)

Régression polynomiale: Flexibilité

Le degré du polynôme affecte directement la flexibilité du modèle.

- Plus d est grand, plus le biais est petit, plus la variance est grande.
- Plus d est petit, plus le biais est grand, plus la variance est petite.

Régression polynomiale: Surapprentissage (1/2)

Inclure un trop haut degré d dans notre modèle peut causer du surapprentissage (dans l'exemple précédent, $d = 10$).

En général, ce degré d est un hyperparamètre, i.e. un paramètre non appris lors de l'apprentissage, que l'on doit préalablement sélectionner sur un échantillon de validation.

Dans les prochain cours, nous étudierons comment bien choisir celui-ci.

Régression polynomiale: Surapprentissage (2/2)

Pour éviter le surapprentissage et déterminer les hyperparamètres d'un modèle "sans tricher", l'idée générale est de calculer la performance du modèle appris sur de nouvelles observations contenues dans l'échantillon de validation.

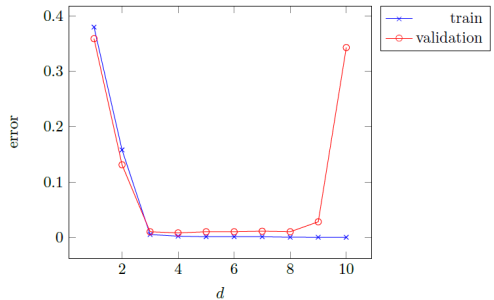
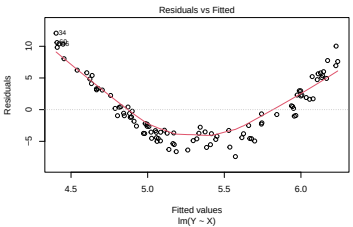


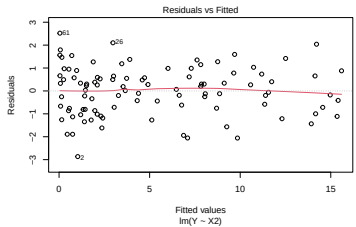
Figure 20: Extrait du livre Shalev-Shwartz and Ben-David (2014)

En pratique

- En pratique on n'utilise que X_j^1 pour débuter, à moins d'avoir une forte raison de croire qu'il est nécessaire d'introduire de la non linéarité.
- En statistique classique, on peut analyser les résidus afin de décider si on intègre des termes polynomiaux.

```
> plot(fit)
```





- En *machine learning*, il est commun d'entraîner plusieurs modèles puis de choisir le meilleur grâce à la validation croisée, une approche générale de sélection de modèles.

Classification linéaire (1/2)

Chapitre 4 du livre de référence James et al. (2013).

Jusqu'à présent, nous avons vu comment modifier les variables explicatives $\mathbf{X}$ afin d'inclure des variables catégorielles ainsi que de capturer des effets non linéaires (intéractions et régression polynomiale).

Un problème standard en science des données, notamment en statistique ou en actuariat, est la classification en groupes discrets et disjoints: Y est une variable qualitative.

Classification linéaire (2/2)

Dans cette courte section, nous introduirons comment modifier le modèle afin de respecter le support de la sortie Y ($\mathbb{R}^+, \mathbb{N}, \{0,1\}$).

On apelle souvent ces modèles les modèles linéaires généralisés ou *Generalized Linear Models* (GLM).

On peut modifier la sortie du modèle linéaire à l'aide d'une fonction lien g:
$g(E[Y|\mathbf{X}]) = \mathbf{X}^T\beta$.

Classification linéaire binaire (1/3)

Supposons que l'on cherche à prédire si un animal est un Chat ou un Chien (Y) en utilisant sa taille (X_1) et son poids (X_2).

En pratique les classes sont représentées par des nombres entiers:

$$Y = \begin{cases} \text{Chat} \\ \text{Chien} \end{cases} = \begin{cases} 0 \\ 1 \end{cases} .$$

Classification linéaire binaire (2/3)

Ici, on aimerait pouvoir prédire le pourcentage de chance/probabilité que l'animal soit un Chat ou un Chien.

Etant donné que $Y \in \{0, 1\}$ est représentée numériquement, **pourquoi ne pas simplement utiliser la régression linéaire** ?

En utilisant un modèle linéaire sur Chat $= 0$ et Chien $= 1$, il existera des combinaisons de taille et poids telles que la sortie ne sera pas entre 0 et 1; donc difficilement interprétable comme une probabilité.

Classification linéaire binaire (3/3)

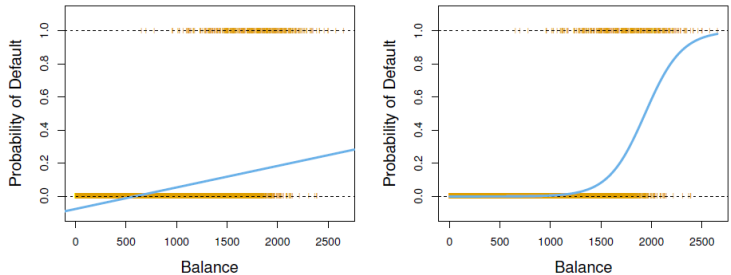


FIGURE 4.2. *Classification using the* `Default` *data.* Left: *Estimated probability of* `default` *using linear regression. Some estimated probabilities are negative! The orange ticks indicate the 0/1 values coded for* `default` *(*`No` *or* `Yes`*).* Right: *Predicted probabilities of* `default` *using logistic regression. All probabilities lie between* 0 *and* 1.

Figure 21: Extrait du livre James et al. (2013)

Régression Logistique

On cherche donc à prédire le vecteur de probabilités (ou distribution de probabilité):

$$\begin{pmatrix} \mathbb{P}(Y=1|\mathbf{X}) \\ \mathbb{P}(Y=0|\mathbf{X}) \end{pmatrix} = \begin{pmatrix} \mathbb{P}(Y=1|\mathbf{X}) \\ 1-\mathbb{P}(Y=1|\mathbf{X}) \end{pmatrix}.$$

Ainsi, dans le cas binaire ($Y \in \{0,1\}$), il nous suffit d'estimer $\mathbb{P}(Y=1|\mathbf{X})$ à l'aide de notre modèle.

Et, on sait qu'il est inapproprié de le faire avec un modèle de régression linéaire standard de type $\mathbb{P}(Y=1|\mathbf{X}) = \mathbf{X}^T\beta$.

Régression Logistique Simple (1/3)

Supposons ici $\mathbf{X} = X_1$.

On sait qu'il est inapproprié d'estimer $\mathbb{P}(Y = 1|X_1)$ à l'aide d'un modèle linéaire standard $\mathbb{P}(Y = 1|X_1) = \beta_0 + \beta_1 X_1$.

Pour remédier à cette situation, nous allons donc essayer d'apprendre $\mathbb{P}(Y = 1|X) = g(\beta_0 + \beta_1 X_1)$, où g est une fonction de lien nous permettant de se ramener à une prédiction de $\mathbb{R}$ dans $[0, 1]$.

Ainsi, $g^{-1}(\mathbb{P}(Y = 1|X)) = \beta_0 + \beta_1 X_1$ est linéaire.

Régression Logistique Simple (2/3)

- Dans la régression logique, g est la fonction logistique ou *sigmoid*: $\forall a \in \mathbb{R}$,

$$g(a) = \frac{e^a}{1+e^a} = \frac{1}{1+e^{-a}} \quad ,$$

- Nous avons donc une probabilité prédite de la forme:

$$\mathbb{P}(Y = 1|X_1) = \frac{e^{\beta_0+\beta_1 X_1}}{1+e^{\beta_0+\beta_1 X_1}} \in [0,1] \quad ,$$

- Et conséquemment:

$$\mathbb{P}(Y = 0|X_1) = 1 - \mathbb{P}(Y = 1|X_1) = \frac{1}{1+e^{\beta_0+\beta_1 X_1}} \quad .$$

Régression Logistique Simple (3/3)

Les coefficients β_0 et β_1 sont estimés en maximisant la vraisemblance des n observations de l'échantillon d'apprentissage:

$$\begin{aligned}\hat{\boldsymbol{\beta}} &= \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \prod_{i=1}^{n} p(y_i; \boldsymbol{\beta}) \\ &= \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \prod_{i=1}^{n} \mathbb{P}(Y_i = 1|x_{i,1})^{y_i} \, \mathbb{P}(Y_i = 0|x_{i,1})^{1-y_i} \\ &= \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \prod_{i=1}^{n} \left(\frac{e^{\beta_0+\beta_1 x_{i,1}}}{1+e^{\beta_0+\beta_1 x_{i,1}}} \right)^{y_i} \left(\frac{1}{1+e^{\beta_0+\beta_1 x_{i,1}}} \right)^{1-y_i}\end{aligned}$$

Il n'existe généralement pas d'expression analytique pour $\boldsymbol{\beta}$. Des algorithmes d'optimisation, de type quasi-Newton, sont alors utilisés.

En pratique (1/2)

Pour prédire une nouvelle observation $X_1 = x_0$ nous calculons:

$$\hat{\mathbb{P}}(Y = 1|X_1 = x_0) = \frac{e^{\beta_0+\beta_1 x_0}}{1 + e^{\beta_0+\beta_1 x_0}} \in [0, 1] \quad .$$

Pour obtenir la classe prédite, on calcule:

$$\hat{Y} = 1_{\hat{\mathbb{P}}(Y=1|X_1=x0) > s}$$

où s est un seuil à déterminer (par exemple, $s = 0.5$).

En pratique (2/2)

```
> fit <- glm(Y ~ X1, family = binomial)
> x0 <- 2
> prob <- predict(fit, newdata = data.frame(X1 = x0),
                  type = 'response')
> pred <- rep(0, length(x0))
> pred[prob > 0.5] = 1
```

Régression Logistique Multiple

Il est simple d'étendre le modèle de régression logistique simple (un seul prédicteur, $\mathbf{X} = X_1$) à un modèle utilisant p prédicteurs, $\mathbf{X} = (X_1, X_2, \cdots, X_p)$:

$$\mathbb{P}(Y = 1|\mathbf{X}) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_p X_p}}{1 + e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_p X_p}}$$

Comme pour le modèle simple, on maximise la vraisemblance d'une loi de Bernoulli des n observations à l'aide d'algorithmes d'optimisation pour estimer $\boldsymbol{\beta} = (\beta_0, \beta_1, \cdots, \beta_p)$.

Classification linéaire multi-classes

Un deuxième problème apparaît est lorsque Y peut appartenir à plus de deux classes.

Prenons:

$$Y = \begin{cases} \text{Chat} = 0 \\ \text{Chien} = 1 \\ \text{Lapin} = 2 \end{cases}$$

Dans cette situation, la régression linéaire traitera "Chien" comme étant une catégorie intermédiaire entre "Lapin" et "Chat", i.e. une interpolation entre les deux.
Cependant, aucune relation d'ordre n'existe entre ces trois catégories, il n'y a donc pas de raison de traiter "Chat" comme étant plus loin de "Lapin" que de "Chien".

Régression Logistique Multinomiale (1/2)

Supposons que nous avons K catégories pour Y, i.e. $Y \in \{1, \cdots, K\}$.

En quelque sorte, la régression logistique introduite permet de modéliser $Y \sim Bern(p)$ où $Y \in \{0, 1\}$.

Si nous avons plus de deux catégories pour Y, nous souhaitons modéliser $Y \sim Mult(p_1, ...p_K)$.

Régression Logistique Multinomiale (2/2)

Encore une fois, l'extension est plutôt simple (surtout si on saute par-dessus les mathématiques).

$\forall\ k \in \{1, \cdots, K-1\}$,

$$\mathbb{P}(Y = k|\mathbf{X}) = \frac{e^{\beta_{k,0}+\beta_{k,1}X_1+\beta_{k,2}X_2+\ldots+\beta_{k,p}X_p}}{1+\sum_{l=1}^{K-1} e^{\beta_{k,0}+\beta_{k,1}X_1+\beta_{k,2}X_2+\ldots+\beta_{k,p}X_p}}$$

et donc,

$$\mathbb{P}(Y = K|\mathbf{X}) = \frac{1}{1+\sum_{l=1}^{K-1} e^{\beta_{k,0}+\beta_{k,1}X_1+\beta_{k,2}X_2+\ldots+\beta_{k,p}X_p}}$$

Dans ce cas, la classe prédite $\hat{Y} = \max_{k\in\{1,\cdots,K\}} \mathbb{P}(Y = k|\mathbf{X})$.

En pratique

```
> library(nnet)
> fit <- multinom(Y ~ X1 + X2, data = data)
> x0 <- c(2, 1)
> prob <- predict(fit, newdata = data.frame(X1 = x0[1], X2 = x0[2]),
                  type = 'probs')
> pred <- which(prob == max(prob))
```

Classification linéaire: Conclusion

- On doit s'assurer que R/Python comprenne que la variable Y est une catégorie.

```
> Y = as.factor(Y)
```

- Lorsqu'on effectue une tâche de classification, il faut comprendre que la sortie est un/des score(s) $\in [0,1]$, estimant une/des probabilité(s).
- Les scores prédits nous permettent d'évaluer l'incertitude/la confiance du modèle dans la prédiction.
- Plus généralement, les différents types de modèles de *machine learning* comprennent à la fois un algorithme spécifique pour les tâches de classification ainsi qu'un algorithme spécifique pour les tâches de régression. Par exemple, en Python (librairie `scikit-learn`):
  - Arbre de décision: `DecisionTreeRegressor`, `DecisionTreeClassifier`,
  - Forêt aléatoire: `RandomForestRegressor`, `RandomForestClassifier`.

Modèles linéaires: Conclusion

- Ce modèle est de moins en moins utilisé en raison des algorithmes de *machine learning* plus complexes donnant généralement une meilleure précision.
- Ce modèle devrait toujours être le modèle de base, puisqu'il permet de donner une intuition sur les relations entre les variables.
- De plus, ce modèle est simple, donc interprétable, et permet notamment de faire de l'inférence sous certaines hypothèses.
- Afin de sélectionner les variables du modèle ou ses hyperparamètres (comme le degré d en régression polynomiale), la régression Lasso ainsi que le mécanisme de validation croisée aident grandement.

Pour le prochain cours

- Lecture: Chapitre 3 et 4 du manuel de référence (James et al. (2013)),
- Lab: 3.6.5, 3.6.6, 4.7.1, 4.7.2,
- Exercises: 3.7.2, 3.7.3, 3.7.4,
- Préparation: Chapitre 6 du manuel de référence (James et al. (2013)).

Références

Charpentier, A. (2023). *Insurance: biases, discrimination and fairness*. Springer Verlag.

James, G., Witten, D., Hastie, T., Tibshirani, R., et al. (2013). *An introduction to statistical learning*, volume 112. Springer.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, USA.