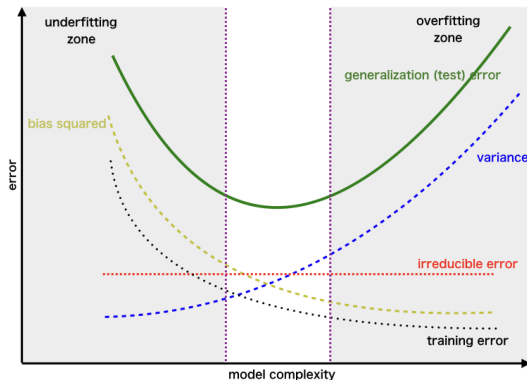


# 1 Grundlagen

- Grundlagen
  - $Y = f(x) + \epsilon$
  - $Y$  = Zielgröße,  $f()$  = unbekannte/wahre Beziehung,  $X$  = Prädiktoren,  $\epsilon$  Nicht reduzierbarer Fehler
  - $\hat{Y} = \hat{f}(X) + \epsilon$
  - $\hat{Y}$  = Schätzung der Zielgröße,  $\hat{f}$  = Schätzung des Modells
  - Ziel: Möglichst genaue Schätzung finden
- Ziel:
  - Prediction (Vorhersage von Werten)
  - Inference (Ursachenanalyse, wie wirken sich Änderungen aus)
- Bias-Variance Tradeoff
  - Bias: Fehler des Modells durch falsche Annahmen der Realität
  - Variance: Fähigkeit des Modells auf anderen Subsets gleich gute Modelle zu erzeugen bzw. Fehleranfälligkeit bei Änderungen in Trainingsdaten
  - TrainingsError: Wird immer kleiner, da Modell sich immer besser anpasst
  - TestError: Wird erst kleiner, steigt dann aber wieder (Overfitting)
  - Nichtreduzierbarer Error: Bleibt immer gleich (Messfehler etc.)



# 2 Regression

- Modellgüte:
  - Schätzung der Parameter  $\beta_0$  und  $\beta_1$  über kleinste Quadrate
    - $\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$
    - $\hat{\beta}_0 = \bar{y} - \beta_1 \bar{x}$
    - Erwartungstreue:  $E(\hat{\beta}_0) = \beta_0$  und  $E(\hat{\beta}_1) = \beta_1$
  - $RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$

- $RSE = \sqrt{\frac{1}{n-2} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$  = Residual Standard Error (Zielgröße weicht im Durchschnitt RSE Einheiten von der Regressionsgeraden ab)
- $R^2 = 1 - \frac{RSS}{TSS}$ , je größer desto besser  $0 \leq R^2 \leq 1$  (Var(Zielwert) wird durch  $R^2\%$  der Prädiktoren erklärt)
- $R^2_{adj} = 1 - \frac{(1-R^2)(N-1)}{N-p-1}$  Adjustiert mit Anzahl der Prädiktoren
- Standardfehler und Intervalle
  - $Var(\epsilon) = \frac{1}{n-2} \sum_{i=1}^n (y_i - \hat{y}_i)^2$
  - $(SE(\hat{\beta}_0))^2 = Var(\epsilon) * (\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^n (x_i - \bar{x})^2})$
  - $(SE(\hat{\beta}_1))^2 = \frac{Var(\epsilon)}{\sum_{i=1}^n (x_i - \bar{x})^2}$
  - Intervallschätzung:  $[\hat{\beta}_1 - 2 * SE(\hat{\beta}_1), \hat{\beta}_1 + 2 * SE(\hat{\beta}_1)]$
  - Interpret: Aus 100 Proben liegt  $\beta_1$  in 95 Fällen im Intervall
  - 95%-Konfidenzintervall: bezieht sich auf den durchschnittlichen Y Wert
  - 95%-Prognoseintervall: bezieht sich auf den konkreten Wert Y von Ausgangswerten  $X_1 \dots$
- t-Test
  - $H_0: \beta_i = 0; H_1: \beta_i \neq 0$
  - $X_i$  Hat keinen Einfluss auf Y
  - $t = \frac{\hat{\beta}_1}{SE(\hat{\beta}_1)}$
  - Einfach den p-Wert ablesen, wenn  $<$  als bspw. 0.05 dann  $H_0$  verwerfen
- F-Test
  - $H_0: \beta_0 = \beta_1 = \dots = 0$
  - $F = \frac{(TSS - RSS)/p}{RSS/(n-p-1)}$
  - Alle Prädiktoren haben keinen Einfluss auf Y
- Qualitative Prädiktoren:
  - Prädiktoren mit 2 Ausprägungen:**
    - DummyVariable aka 0(No) oder 1 (Yes)
    - Achte auf Normalausprägung von R**
    - $\hat{y} = \beta_0 + \beta_1 * x_i$
    - Koeffizient  $\beta_1$  kürzt sich je nach Ausprägung raus
  - Prädiktoren mit k Ausprägungen:**
    - Erstelle  $k - 1$  Dummyvariablen
    - Andere ist Normalzustand
- Interaktionseffekte:
  - Synergieeffekte zwischen zwei oder mehreren Variablen
  - $\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \epsilon$
  - Auswirkung erkennen durch Umformung:
    - $\hat{y} = \beta_0 + \beta_2 x_2 + (\beta_1 + \beta_3 x_2) * x_1$
  - Erhöht man  $x_1$  um eine Einheit erhöht sich  $\hat{y}$  um  $\beta_1 + \beta_3 x_2$  Einheiten
  - $x_1$  moderiert  $x_2$  und Vice versa
  - Signifikanz über p-value feststellen
  - Interaktion zwischen Quali und Quanti Variablen:

- Kürzt sich komplett raus (wenn 0) oder ist \*1 (wenn 1)

# 3 Klassifikation

- Grundlagen:**
  - Klassifikation ist Zuordnung von bedingten Wahrscheinlichkeiten anhand von Ausprägungen
  - $P(Y = k | X = x)$
  - Klasse mit höchster Wahrscheinlichkeit wird gewählt
  - Bpsw: Y = 1 (Yes) und 0 (No)
    - Lineare Regression, die die Wahrscheinlichkeitsausgibt **funktioniert nicht!**
    - Reg. Gerade wird unter 0 fitten und auch nicht bis 1
  - Funktion finden, die  $0 \leq p(x) \leq 1$  für alle X
- Logistic Regression:**
  - $p(x) = \frac{e^{\beta_0 + \beta_1 * X}}{1 + e^{\beta_0 + \beta_1 * X}}$
  - $0 \leq p(x) \leq 1$ , da immer  $e^x \geq 0$  und  $\frac{X}{1+X} \leq 1$
  - Odds:
    - $\frac{p(x)}{1-p(x)} = e^{\beta_0 + \beta_1 * X}$
    - $0 \leq odds \leq \infty$
    - Möglichkeit Wahrscheinlichkeit anzugeben:
      - Odds =  $\frac{4}{3}$  -> "Siegchancen stehen 4 zu 3"
    - Umrechnung:
      - $p = \frac{odds}{1+odds} = \frac{e^{logit}}{1+e^{logit}}$
      - $odds = \frac{p}{1-p} = e^{logit}$
  - Logits:
    - $\log(\frac{p(x)}{1-p(x)}) = \beta_0 + \beta_1 * X$
    - $-\infty \leq logits \leq \infty$
    - Logits hängen Linear von X ab
- Schätzung der Koeffizienten**
  - Schätzung von  $\beta_0$  und  $\beta_1$  über Maximum-Likelihood
  - $l(\beta_0, \beta_1) = \prod_{i: y_i=1} p(x_i) * \prod_{i: y_i=0} (1 - p(x_i))$
  - Funktion wird maximiert = da wo Likelihood am größten ist, sind Parameter am besten
- Validierung:**
  - Gleiche Werte/ Tests, wie bei lineare Regression
- Confounding:**
  - Zusammenhang zwischen zwei Prädiktoren
  - Trugschluss, wenn man nur einzelne Variable in Abhängigkeit zu Y betrachtet
  - In der Gesamtmasse wird es dann gut
  - Confounder beeinflusst gleichzeitig Zielgröße und anderen Prädiktor

# 4 Resampling

- Grundlagen:**
  - TestError wird mit ungesehenen Daten berechnet.

- Meist aber keine vorhanden -> TestError muss über Trainingsdaten geschätzt werden
- Verfahren für Regression/Klassifikation anwendbar
- **Einfache Validierung:**
  - Teile Trainingsdaten in zwei gleiche große Mengen auf
  - Menge1: für Training / Menge2: für Testing
  - $MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$
  - MSE variiert stark in Abhängigkeit der Trainingsmenge
  - Modelle sollten möglichst mit hoher Trainingsmenge trainiert werden
  - Hoher Bias, geringe Varianz
- **LeaveOneOut Validierung:**
  - n-Modelle werden mit n-1 Datensätze trainiert
  - Letzter Datensatz ist Testdatensatz
  - Durchschnitt aller MSEs ist finaler Wert
  - $CV_{(n)} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$
  - Vorteil: Größere Trainingsmenge
  - Nachteil: Hoher Rechenaufwand für n Modelle, größere Varianz als bei einfacher Val, durch Autokorrelation
  - geringer Bias, hohe Varianz
- **k-fold Validierung**
  - Teile Datensatz in k gleich große Mengen (5, 10..)
  - Trainiere k Modelle, wobei immer eine Teilmenge als Test genutzt wird
  - $CV_{(k)} = \frac{1}{k} \sum_{i=1}^k \frac{1}{n} \sum_{j=1}^n (y_j - \hat{y}_j)^2$
  - geringere Streuung des MSEs
  - Guter Kompromiss zwischen Rechenaufwand und guter Abschätzung
- Für Klassifikation wird der Anteil falsch klassifizierter Objekte genommen!

## 5 Modellauswahl

- p (# Prädiktoren), n (# Beobachtungen)
- Steigende Varianz und sinkender Prognosegüte, wenn p -> n
- Hohe Anzahl an Prädiktoren: Steigende Wahrscheinlichkeit von Multikollinearität (Abhängige Prädiktoren) und unnötige Komplexität
- Wenn  $p \geq n$ , ist Methode der kleinsten Quadrate gar nicht möglich
- **Best Subset Selection:**
  - Berechne alle Modelle mit k Parametern (Speicher das beste (Min. RSS oder Max.  $R^2$ ))
  - Berechne für k = 0,1,2 .. x alle besten Modelle
  - Wähle beste aus Besten aus (siehe unten)
  - Hoher Rechenaufwand ( $2^p$  Modelle), Gefahr von Overfit-

ting, findet optimales Modell

- **Forward Stepwise Selection:**
  - Berechne bestes Modell mit 0 Prädiktoren, dann mit 1, etc.
  - Wähle für k immer das beste Modell und gehe von dort aus weiter
  - Wähle beste aus Besten aus
  - Weniger Rechenaufwand ( $p^2$  Modelle), Overfitting unwahrscheinlich, findet nur lokales Optimum
- **Backward Stepwise Selection:**
  - Wie Forward Stepwise Selection nur rückwärts
  - Starte mit Modell mit k Prädiktoren und nimm immer den schlechtesten Weg
  - Wähle beste aus Besten aus
  - Weniger Rechenaufwand ( $p^2$  Modelle), Overfitting unwahrscheinlich, findet nur lokales Optimum
- **Bestes Modell?**
  - $R^2$  und RSS ist nicht aussagekräftig, da es mit steigender Anzahl an Prädiktoren besser wird.
  - Testfehler indirekt messbar durch ( $C_p, AIC, BIC$  und  $R_{adj}^2$ ) oder direkt durch Kreuzvalidierung
  - Anwendbar bei linearen Modellen
  - $RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$
  - n: Anzahl der Beobachtungen, d: Anzahl der Prädiktoren
  - $\sigma^2 = Var(e) = \frac{1}{n-p-1} * RSS$
  - Mallows  $C_p = \frac{1}{n} (RSS + 2d\sigma^2)$
  - $AIC = \frac{1}{n\sigma^2} (RSS + 2d\sigma^2)$  (Proportional zu  $C_p$ )
  - $BIC = \frac{1}{n\sigma^2} (RSS + \ln(n)d\sigma^2)$  (Höheres Gewicht auf Strafterm für viel Prädiktoren)
  - $R_a^2 = R^2 - \frac{p}{n-p-1} (1 - R^2)$  kann bei Hinzunahme einer Variable auch kleiner und oder negativ werden

## 6 R - Hilfe

- **Grundlagen:**
  - *set.seed(X)* Setzt Seed für random Number Generator
  - *c(1,2,3,4)* Vektor mit Zahlen 1-4
  - *rep(1:10,100)* Erzeugt Vektor mit zahlen 1-10, 100 mal
  - *rnorm(100)* Erzeugt Vektor mit 100 normalverteilten Werten
  - *runif(X,a,b)* Erzeugt X Zufallszahlen zwischen a und b
  - *df[2,3]* Greift auf Element der 2.Reihe und 3.Spalte des DFs zu
  - *df[, -3]* Entfernt 3. Spalte
  - *head()* Zeigt erste X Zeilen von DF an
  - *summary()* gibt Zusammenfassung von Modellen (DF, Modelle etc.)

- *table(X,Y)* gibt Tabelle bzw. Konfusionsmatrix aus
- *nrow(X)* gibt Anzahl der Zeilen in X
- *sample(n,n/2)* gibt 50% Subsetvektor
- *is.na(X)* findet/zählt null Values
- *na.omit(X)* entfernt null Values
- *var(X), mean(X)* Varianz und Durchschnitt von X
- **Modelle:**
  - *lm(A poly(C,2) + B : D, data = ..., subset =)* Lineare Regression für A mit Interaktivität von BD und C mit Exponent 2
  - *glm(A B + C, data = ..., family = binomial, subset =)* Logistic Regression Modell
  - *predict(Modell, DataFrame, interval =, type =)*
    - \* DF: *data.frame(x1 = c(2), x2 = c(3))*
    - \* *interval* Konfidenzintervall(confidence), Prognoseintervall(prediction)
    - \* *type* Wahrscheinlichkeit(response), Loggits(ohne Angabe)
  - *coef()* Zeigt Koeffizienten des Modells
  - *confint()* Zeigt Konfidenzintervalle für Koeff.
  - *chisq.test(X,Y)* Macht ChiquadratTest auf Unabhängigkeit von X und Y ( $H_0$ : Merkmale sind unabhängig)
- **Validierung:**
  - *mean((pred,rue - predict(Modell, Datenatz))[-train]^2)* Berechnet die mittlere quadratische Abweichung
  - Bei Klassifikation vorher Wahrscheinlichkeiten umwandeln in Quali-Werte
  - aus dem *boot* package
  - *cv.glm(Data, Modell)\$delta[1]* LeaveOneOut ( $\$delta[1]$  liefert nur ersten Wert also MSE)
  - *cv.glm(Data, Modell, k = X)\$delta[1]* k-fold mit k Folds
  - *regsubsets(Y, data =, nvmax =, method =)* Subset-Selection (nvmax = Anzahl Attribute, methode= forward/backward/none)
- **Plots:**
  - *par(mfrow = c(2,2))* Zeigt 2x2 Matrix mit Plots
  - *pairs()* Zeigt Pärchenplott aller quantitativer Variablen
  - *plot(x,y,xlab =, ylab =)* Zeigt X/Y Plot zweier Variablen mit Achsenbeschriftung
  - *plot(modell)* Zeigt Plots zu Residuen etc.
  - *hist()* Zeigt Histogramm
  - *points()* Erzeugt Punkt im Plot
  - *abline(Modell, col = "red")* Zeigt Regressionslinie
  - *qplot()* aus *ggplot2* für quickplot