Erick Marcellino Pranata – 210711155

Jacklyn Fionadewi Suseno – 210711210

Elluy Gabriel Panambe – 210711306

Alfa Nada Yulaswara – 210711378

Tugas Kelompok Klasifikasi

1. Pengertian Dasar Metode Ensemble

**Metode Ensembel**

Metode Ensemble adalah teknik lanjutan dalam pembelajaran mesin yang mengkombinasikan prediksi dari beberapa model untuk menghasilkan prediksi yang lebih akurat dan kuat daripada yang bisa dicapai oleh model tunggal. Pendekatan ini memanfaatkan kekuatan dan mengurangi kelemahan berbagai model dengan menggabungkan hasil mereka. Beberapa metode ensemble yang umum digunakan meliputi:

* **Bagging (Bootstrap Aggregating)**: Metode ini melibatkan pelatihan beberapa model secara independen pada subset acak yang berbeda dari data pelatihan (diperoleh melalui bootstrap) dan mengambil rata-rata prediksi mereka untuk tugas regresi atau mengambil suara mayoritas untuk tugas klasifikasi. Contoh populer dari teknik ini adalah Random Forest.
* **Boosting**: Boosting membangun model secara berurutan, di mana setiap model baru mencoba memperbaiki kesalahan yang dibuat oleh model sebelumnya. Dengan cara ini, boosting lebih fokus pada contoh-contoh yang sulit diprediksi. Contoh algoritma boosting termasuk AdaBoost, Gradient Boosting Machines (GBM), dan CatBoost.
* **Stacking**: Metode ini melibatkan pelatihan beberapa model (disebut pelajar dasar) dan kemudian menggabungkan prediksi mereka menggunakan model meta. Prediksi dari pelajar dasar digunakan sebagai fitur untuk model meta, yang kemudian membuat prediksi akhir.
* **Voting**: Dalam metode ini, beberapa model dilatih, dan prediksi mereka dikumpulkan melalui suara mayoritas (untuk klasifikasi) atau rata-rata (untuk regresi). Voting bisa bersifat hard (kelas mayoritas) atau soft (probabilitas tertimbang).

1. Konsep Dasar Algoritma CatBoost

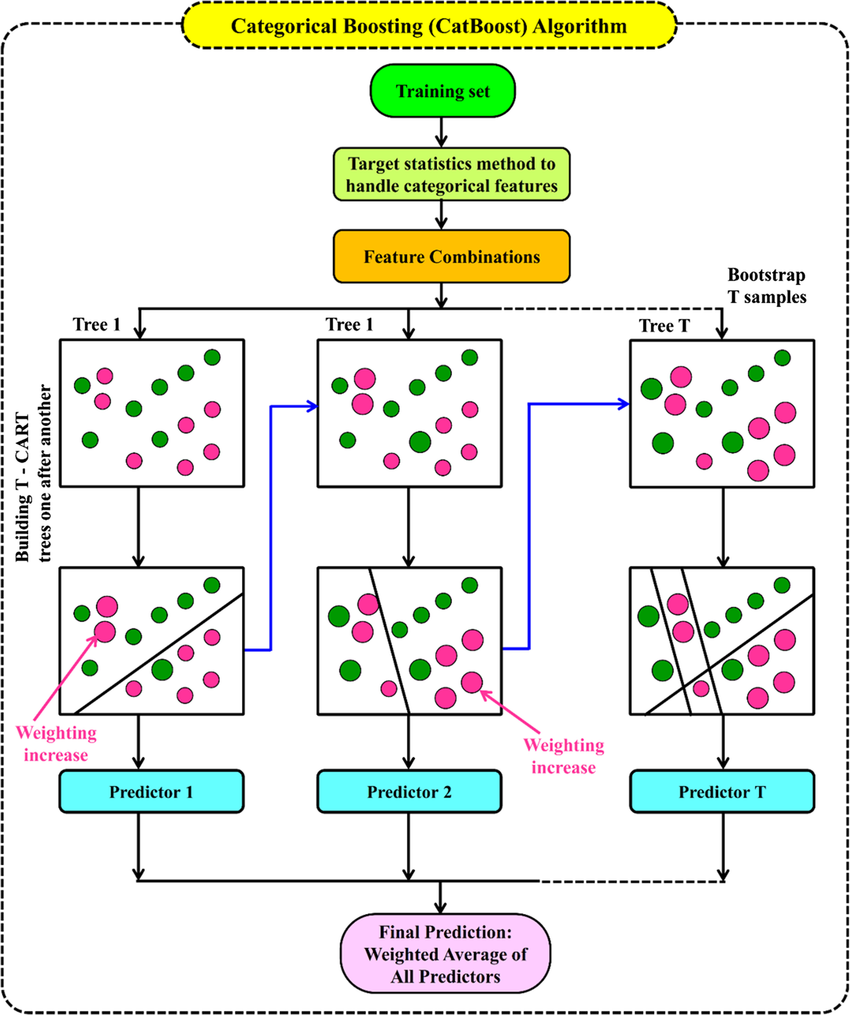
**CatBoost**



CatBoost (Categorical Boosting) adalah algoritma peningkatan canggih yang dikembangkan oleh Yandex. Ini dirancang untuk menangani fitur kategorikal secara efisien dan memberikan kinerja unggul untuk berbagai jenis tugas pembelajaran mesin. Berikut beberapa konsep dan fitur utama CatBoost:

* **Handling Categorical Features:** CatBoost sangat efektif dalam menangani data kategorikal. Ia menggunakan metode statistik target untuk mengubah fitur kategorikal menjadi nilai numerik, sehingga mengurangi kebutuhan akan pra-pemrosesan dan pengkodean yang ekstensif.
* **Ordered Boosting:** Tidak seperti algoritme peningkatan tradisional yang dapat mengalami pergeseran prediksi karena cara data digunakan untuk pelatihan dan validasi, CatBoost menggunakan peningkatan terurut. Metode ini memastikan bahwa prediksi setiap titik data hanya didasarkan pada titik data sebelumnya, mencegah kebocoran label dan meningkatkan generalisasi.
* **Symmetric Trees:** CatBoost membangun pohon simetris, artinya setiap pemisahan terjadi pada tingkat yang sama untuk semua pohon. Hal ini menghasilkan waktu pelatihan dan prediksi yang lebih cepat, serta interpretasi model yang lebih baik.
* **Efficient Training:** CatBoost menggunakan teknik pengoptimalan tingkat lanjut untuk membuat proses pelatihan lebih efisien, menangani kumpulan data besar dengan dimensi tinggi secara efektif.
* **Robustness to Overfitting:** CatBoost menggabungkan beberapa mekanisme untuk mengatasi overfitting, seperti regularisasi berbasis gradien dan penghentian dini.

Alur Kerja Algoritma CatBoost



Alur kerja algoritma CatBoost seperti yang diilustrasikan pada diagram dapat diringkas sebagai berikut:

1. **Persiapan Set Pelatihan:** Algoritma dimulai dengan set pelatihan yang mencakup fitur kategorikal dan numerik.
2. **Menangani Fitur Kategoris:** Metode statistik target diterapkan untuk menangani fitur kategoris dengan mengubahnya menjadi representasi numerik. Selain itu, kombinasi fitur dibuat untuk meningkatkan kemampuan model dalam menangkap pola kompleks.
3. **Membangun Pohon Secara Berurutan:** Urutan pohon T (atau prediktor) dibangun satu demi satu. Setiap pohon dibangun menggunakan subset data pelatihan yang diperoleh melalui bootstrapping.
4. **Pembobotan dan Koreksi Kesalahan:** Selama pelatihan setiap pohon, kejadian yang salah diprediksi oleh pohon sebelumnya diberi bobot yang lebih besar. Hal ini memfokuskan pohon baru untuk memperbaiki kesalahan ini, sehingga meningkatkan akurasi model secara keseluruhan.
5. **Prediksi Akhir:** Prediksi akhir dibuat dengan mengambil rata-rata tertimbang dari semua prediktor. Agregasi keluaran beberapa model ini membantu mencapai prediksi yang lebih akurat dan kuat.

Dengan menggabungkan prinsip-prinsip ini, CatBoost memberikan solusi yang sangat efisien dan efektif untuk tugas-tugas pembelajaran mesin, terutama ketika menangani data kategorikal.

1. Contoh implementasi CatBoos

#library(devtools)

#options(devtools.install.args = c("–no-multiarch", "–no-test-load"))

#install.packages("C:/Users/Gabriel/Downloads/catboost-R-windows-x86\_64-1.2.5.tgz", repos = NULL, type = "source", INSTALL\_opts = c("–no-multiarch", "–no-test-load"))

#install.packages("caret")

#install.packages("titanic")

library(catboost)

library(caret)

library(titanic)

# load data

set.seed(1)

idx=sample(1:nrow(iris),nrow(iris)\*.7)

train=iris[idx,]

test=iris[-idx,]

fit\_control <- caret::trainControl(

method = "cv",

number = 3,

search = "random",

classProbs = TRUE

)

# set grid options

grid <- expand.grid(

depth = c(4, 6, 8),

learning\_rate = 0.1,

l2\_leaf\_reg = 0.1,

rsm = 0.95,

border\_count = 64,

iterations = 10

)

model <- caret::train(

x = train[,-5],

y = train[,5],

method = catboost.caret,

metric = "Accuracy",

maximize = TRUE,

preProc = NULL,

tuneGrid = grid,

tuneLength = 30,

trControl = fit\_control

)

table(test$Species,predict(model,test))

# Pertama, data Titanic diubah menjadi format data frame:

data <- as.data.frame(as.matrix(titanic::titanic\_train), stringsAsFactors=TRUE)

# Kemudian, nilai NA pada kolom Age diisi dengan nilai yang paling sering muncul di kolom tersebut:

age\_levels <- levels(data$Age)

most\_frequent\_age <- which.max(table(data$Age))

data$Age[is.na(data$Age)] <- age\_levels[most\_frequent\_age]

# Kolom yang tidak diperlukan untuk pelatihan model dihapus, dan data dibagi menjadi fitur (x) dan target (y):

drop\_columns = c("PassengerId", "Survived", "Name", "Ticket", "Cabin")

x <- data[,!(names(data) %in% drop\_columns)]

y <- data[,c("Survived")]

# Kontrol pelatihan untuk cross-validation diatur menggunakan caret:

fit\_control <- caret::trainControl(

method = "cv",

number = 3,

search = "random",

classProbs = TRUE

)

# Parameter grid untuk model CatBoost ditentukan:

grid <- expand.grid(

depth = c(4, 6, 8),

learning\_rate = 0.1,

l2\_leaf\_reg = 0.1,

rsm = 0.95,

border\_count = 64,

iterations = 10

)

# Model CatBoost dilatih menggunakan caret dengan parameter grid yang telah ditentukan:

model <- caret::train(

x = x,

y = as.factor(make.names(y)),

method = catboost.caret,

metric = "Accuracy",

maximize = TRUE,

preProc = NULL,

tuneGrid = grid,

tuneLength = 30,

trControl = fit\_control

)

print(model)

# variable importance

importance <- varImp(model, scale = FALSE)

print(importance)

plot(importance)

**Hasil :**

> #library(devtools)

>

> #options(devtools.install.args = c("–no-multiarch", "–no-test-load"))

>

> #install.packages("C:/Users/Gabriel/Downloads/catboost-R-windows-x86\_64-1.2.5.tgz", repos = NULL, type = "source", INSTALL\_opts = c("–no-multiarch", "–no-test-load"))

>

> #install.packages("caret")

> #install.packages("titanic")

> library(catboost)

> library(caret)

Loading required package: ggplot2

Loading required package: lattice

> library(titanic)

>

> # load data

> set.seed(1)

> idx=sample(1:nrow(iris),nrow(iris)\*.7)

> train=iris[idx,]

> test=iris[-idx,]

> fit\_control <- caret::trainControl(

+ method = "cv",

+ number = 3,

+ search = "random",

+ classProbs = TRUE

+ )

> # set grid options

> grid <- expand.grid(

+ depth = c(4, 6, 8),

+ learning\_rate = 0.1,

+ l2\_leaf\_reg = 0.1,

+ rsm = 0.95,

+ border\_count = 64,

+ iterations = 10

+ )

> model <- caret::train(

+ x = train[,-5],

+ y = train[,5],

+ method = catboost.caret,

+ metric = "Accuracy",

+ maximize = TRUE,

+ preProc = NULL,

+ tuneGrid = grid,

+ tuneLength = 30,

+ trControl = fit\_control

+ )

0: learn: 0.9842096 total: 141ms remaining: 1.26s

1: learn: 0.8463735 total: 141ms remaining: 565ms

2: learn: 0.7495783 total: 142ms remaining: 331ms

3: learn: 0.6742001 total: 143ms remaining: 214ms

4: learn: 0.6074442 total: 143ms remaining: 143ms

5: learn: 0.5457995 total: 144ms remaining: 95.8ms

6: learn: 0.4935719 total: 144ms remaining: 61.9ms

7: learn: 0.4536394 total: 145ms remaining: 36.2ms

8: learn: 0.4079550 total: 145ms remaining: 16.2ms

9: learn: 0.3690324 total: 146ms remaining: 0us

0: learn: 0.9572373 total: 1.05ms remaining: 9.49ms

1: learn: 0.8468853 total: 1.87ms remaining: 7.48ms

2: learn: 0.7501763 total: 2.85ms remaining: 6.64ms

3: learn: 0.6562598 total: 3.84ms remaining: 5.76ms

4: learn: 0.5865871 total: 4.55ms remaining: 4.55ms

5: learn: 0.5283046 total: 5.28ms remaining: 3.52ms

6: learn: 0.4820792 total: 5.99ms remaining: 2.56ms

7: learn: 0.4360216 total: 6.73ms remaining: 1.68ms

8: learn: 0.3994521 total: 7.78ms remaining: 863us

9: learn: 0.3643118 total: 8.56ms remaining: 0us

0: learn: 0.9491403 total: 2.12ms remaining: 19.1ms

1: learn: 0.8255136 total: 6.08ms remaining: 24.3ms

2: learn: 0.7127905 total: 8.24ms remaining: 19.2ms

3: learn: 0.6335239 total: 10.1ms remaining: 15.2ms

4: learn: 0.5603700 total: 12ms remaining: 12ms

5: learn: 0.4980514 total: 14.3ms remaining: 9.53ms

6: learn: 0.4473533 total: 16.2ms remaining: 6.96ms

7: learn: 0.4080424 total: 18.1ms remaining: 4.52ms

8: learn: 0.3667661 total: 19.3ms remaining: 2.14ms

9: learn: 0.3337294 total: 21ms remaining: 0us

0: learn: 0.9366164 total: 597us remaining: 5.38ms

1: learn: 0.8330805 total: 1.14ms remaining: 4.58ms

2: learn: 0.7232014 total: 1.54ms remaining: 3.58ms

3: learn: 0.6503169 total: 1.93ms remaining: 2.89ms

4: learn: 0.5728482 total: 2.31ms remaining: 2.31ms

5: learn: 0.5100108 total: 2.71ms remaining: 1.8ms

6: learn: 0.4551064 total: 3.13ms remaining: 1.34ms

7: learn: 0.4154533 total: 3.55ms remaining: 887us

8: learn: 0.3756054 total: 4.09ms remaining: 454us

9: learn: 0.3429265 total: 4.67ms remaining: 0us

0: learn: 0.9380051 total: 1.07ms remaining: 9.64ms

1: learn: 0.8309043 total: 1.81ms remaining: 7.23ms

2: learn: 0.7379528 total: 2.82ms remaining: 6.58ms

3: learn: 0.6504317 total: 3.91ms remaining: 5.87ms

4: learn: 0.5776358 total: 4.77ms remaining: 4.77ms

5: learn: 0.5152137 total: 5.55ms remaining: 3.7ms

6: learn: 0.4708744 total: 6.29ms remaining: 2.7ms

7: learn: 0.4251012 total: 7.06ms remaining: 1.76ms

8: learn: 0.3864483 total: 8.28ms remaining: 920us

9: learn: 0.3551740 total: 9.08ms remaining: 0us

0: learn: 0.9383656 total: 2.01ms remaining: 18.1ms

1: learn: 0.8061321 total: 3.86ms remaining: 15.4ms

2: learn: 0.7038528 total: 5.46ms remaining: 12.7ms

3: learn: 0.6241490 total: 8.15ms remaining: 12.2ms

4: learn: 0.5510047 total: 9.74ms remaining: 9.74ms

5: learn: 0.4932954 total: 11.6ms remaining: 7.72ms

6: learn: 0.4426992 total: 12.6ms remaining: 5.4ms

7: learn: 0.4001174 total: 14.3ms remaining: 3.56ms

8: learn: 0.3639627 total: 16ms remaining: 1.78ms

9: learn: 0.3283160 total: 17.9ms remaining: 0us

0: learn: 0.9356814 total: 594us remaining: 5.35ms

1: learn: 0.8085192 total: 1.08ms remaining: 4.32ms

2: learn: 0.7064664 total: 1.5ms remaining: 3.5ms

3: learn: 0.6431167 total: 1.93ms remaining: 2.89ms

4: learn: 0.5771164 total: 2.38ms remaining: 2.38ms

5: learn: 0.5116376 total: 2.82ms remaining: 1.88ms

6: learn: 0.4616117 total: 3.24ms remaining: 1.39ms

7: learn: 0.4166586 total: 3.72ms remaining: 930us

8: learn: 0.3755562 total: 4.13ms remaining: 458us

9: learn: 0.3452118 total: 4.47ms remaining: 0us

0: learn: 0.9371422 total: 985us remaining: 8.87ms

1: learn: 0.8148969 total: 1.66ms remaining: 6.65ms

2: learn: 0.7125098 total: 2.36ms remaining: 5.51ms

3: learn: 0.6284014 total: 3.31ms remaining: 4.96ms

4: learn: 0.5590979 total: 4.02ms remaining: 4.02ms

5: learn: 0.4981345 total: 4.9ms remaining: 3.27ms

6: learn: 0.4561460 total: 5.79ms remaining: 2.48ms

7: learn: 0.4162041 total: 6.83ms remaining: 1.71ms

8: learn: 0.3757165 total: 7.91ms remaining: 878us

9: learn: 0.3419775 total: 8.82ms remaining: 0us

0: learn: 0.9364365 total: 2.32ms remaining: 20.9ms

1: learn: 0.8022667 total: 4.31ms remaining: 17.2ms

2: learn: 0.7058004 total: 6.04ms remaining: 14.1ms

3: learn: 0.6307320 total: 6.33ms remaining: 9.49ms

4: learn: 0.5690402 total: 6.94ms remaining: 6.94ms

5: learn: 0.5071240 total: 8.54ms remaining: 5.7ms

6: learn: 0.4528547 total: 10.3ms remaining: 4.43ms

7: learn: 0.4344095 total: 10.6ms remaining: 2.65ms

8: learn: 0.3869270 total: 12.5ms remaining: 1.39ms

9: learn: 0.3532717 total: 14ms remaining: 0us

0: learn: 0.9267782 total: 2.32ms remaining: 20.9ms

1: learn: 0.8051883 total: 4.33ms remaining: 17.3ms

2: learn: 0.7198290 total: 5.84ms remaining: 13.6ms

3: learn: 0.6320911 total: 7.9ms remaining: 11.9ms

4: learn: 0.5678734 total: 9.83ms remaining: 9.83ms

5: learn: 0.5025111 total: 11.8ms remaining: 7.89ms

6: learn: 0.4513918 total: 13.1ms remaining: 5.63ms

7: learn: 0.4066172 total: 14.9ms remaining: 3.73ms

8: learn: 0.3697060 total: 16.7ms remaining: 1.86ms

9: learn: 0.3387834 total: 17.4ms remaining: 0us

>

> table(test$Species,predict(model,test))

setosa versicolor virginica

setosa 15 0 0

versicolor 0 17 0

virginica 0 1 12

>

> # Pertama, data Titanic diubah menjadi format data frame:

> data <- as.data.frame(as.matrix(titanic::titanic\_train), stringsAsFactors=TRUE)

>

> # Kemudian, nilai NA pada kolom Age diisi dengan nilai yang paling sering muncul di kolom tersebut:

> age\_levels <- levels(data$Age)

> most\_frequent\_age <- which.max(table(data$Age))

> data$Age[is.na(data$Age)] <- age\_levels[most\_frequent\_age]

>

> # Kolom yang tidak diperlukan untuk pelatihan model dihapus, dan data dibagi menjadi fitur (x) dan target (y):

> drop\_columns = c("PassengerId", "Survived", "Name", "Ticket", "Cabin")

> x <- data[,!(names(data) %in% drop\_columns)]

> y <- data[,c("Survived")]

>

> # Kontrol pelatihan untuk cross-validation diatur menggunakan caret:

> fit\_control <- caret::trainControl(

+ method = "cv",

+ number = 3,

+ search = "random",

+ classProbs = TRUE

+ )

>

> # Parameter grid untuk model CatBoost ditentukan:

> grid <- expand.grid(

+ depth = c(4, 6, 8),

+ learning\_rate = 0.1,

+ l2\_leaf\_reg = 0.1,

+ rsm = 0.95,

+ border\_count = 64,

+ iterations = 10

+ )

>

> # Model CatBoost dilatih menggunakan caret dengan parameter grid yang telah ditentukan:

> model <- caret::train(

+ x = x,

+ y = as.factor(make.names(y)),

+ method = catboost.caret,

+ metric = "Accuracy",

+ maximize = TRUE,

+ preProc = NULL,

+ tuneGrid = grid,

+ tuneLength = 30,

+ trControl = fit\_control

+ )

0: learn: 0.6532387 total: 32.3ms remaining: 291ms

1: learn: 0.6277171 total: 58.2ms remaining: 233ms

2: learn: 0.5981361 total: 74.8ms remaining: 175ms

3: learn: 0.5714289 total: 92.6ms remaining: 139ms

4: learn: 0.5512899 total: 108ms remaining: 108ms

5: learn: 0.5365430 total: 126ms remaining: 84.2ms

6: learn: 0.5232219 total: 144ms remaining: 61.6ms

7: learn: 0.5105448 total: 162ms remaining: 40.5ms

8: learn: 0.4998955 total: 179ms remaining: 19.9ms

9: learn: 0.4928092 total: 197ms remaining: 0us

0: learn: 0.6532387 total: 21ms remaining: 189ms

1: learn: 0.6282048 total: 32.6ms remaining: 130ms

2: learn: 0.5999280 total: 45.5ms remaining: 106ms

3: learn: 0.5754993 total: 70.9ms remaining: 106ms

4: learn: 0.5541360 total: 94.7ms remaining: 94.7ms

5: learn: 0.5331561 total: 123ms remaining: 82.2ms

6: learn: 0.5167809 total: 151ms remaining: 64.5ms

7: learn: 0.5051385 total: 166ms remaining: 41.6ms

8: learn: 0.4924111 total: 193ms remaining: 21.5ms

9: learn: 0.4844617 total: 205ms remaining: 0us

0: learn: 0.6532387 total: 26.9ms remaining: 243ms

1: learn: 0.6282048 total: 40.6ms remaining: 163ms

2: learn: 0.5999280 total: 54.7ms remaining: 128ms

3: learn: 0.5754993 total: 77.4ms remaining: 116ms

4: learn: 0.5507920 total: 114ms remaining: 114ms

5: learn: 0.5297626 total: 150ms remaining: 100ms

6: learn: 0.5198746 total: 166ms remaining: 71.1ms

7: learn: 0.5086424 total: 179ms remaining: 44.8ms

8: learn: 0.4941782 total: 219ms remaining: 24.3ms

9: learn: 0.4850398 total: 254ms remaining: 0us

0: learn: 0.6537156 total: 16.3ms remaining: 147ms

1: learn: 0.6248384 total: 31.4ms remaining: 126ms

2: learn: 0.5953647 total: 50.1ms remaining: 117ms

3: learn: 0.5704001 total: 67.9ms remaining: 102ms

4: learn: 0.5522672 total: 85.8ms remaining: 85.8ms

5: learn: 0.5367111 total: 105ms remaining: 69.7ms

6: learn: 0.5233947 total: 119ms remaining: 50.9ms

7: learn: 0.5126362 total: 138ms remaining: 34.5ms

8: learn: 0.5011484 total: 161ms remaining: 17.8ms

9: learn: 0.4917685 total: 182ms remaining: 0us

0: learn: 0.6537156 total: 14.1ms remaining: 127ms

1: learn: 0.6248384 total: 27.7ms remaining: 111ms

2: learn: 0.5938073 total: 58ms remaining: 135ms

3: learn: 0.5685011 total: 86.4ms remaining: 130ms

4: learn: 0.5485062 total: 112ms remaining: 112ms

5: learn: 0.5312579 total: 129ms remaining: 85.7ms

6: learn: 0.5163078 total: 155ms remaining: 66.3ms

7: learn: 0.5031349 total: 172ms remaining: 42.9ms

8: learn: 0.4924867 total: 189ms remaining: 21ms

9: learn: 0.4806827 total: 217ms remaining: 0us

0: learn: 0.6537156 total: 14.5ms remaining: 130ms

1: learn: 0.6248384 total: 27.5ms remaining: 110ms

2: learn: 0.5934537 total: 62.7ms remaining: 146ms

3: learn: 0.5670453 total: 97ms remaining: 145ms

4: learn: 0.5468333 total: 114ms remaining: 114ms

5: learn: 0.5267651 total: 143ms remaining: 95.5ms

6: learn: 0.5114611 total: 167ms remaining: 71.7ms

7: learn: 0.4946778 total: 203ms remaining: 50.9ms

8: learn: 0.4824193 total: 242ms remaining: 26.9ms

9: learn: 0.4703705 total: 281ms remaining: 0us

0: learn: 0.6550874 total: 15.5ms remaining: 140ms

1: learn: 0.6299542 total: 30.6ms remaining: 122ms

2: learn: 0.6024134 total: 50.4ms remaining: 118ms

3: learn: 0.5762565 total: 70.1ms remaining: 105ms

4: learn: 0.5576420 total: 88.8ms remaining: 88.8ms

5: learn: 0.5431542 total: 108ms remaining: 72.2ms

6: learn: 0.5298082 total: 122ms remaining: 52.3ms

7: learn: 0.5160254 total: 141ms remaining: 35.2ms

8: learn: 0.5049215 total: 167ms remaining: 18.6ms

9: learn: 0.4950305 total: 194ms remaining: 0us

0: learn: 0.6550874 total: 17.6ms remaining: 158ms

1: learn: 0.6299542 total: 31.1ms remaining: 124ms

2: learn: 0.6024134 total: 62.7ms remaining: 146ms

3: learn: 0.5754979 total: 92.3ms remaining: 138ms

4: learn: 0.5620408 total: 102ms remaining: 102ms

5: learn: 0.5432074 total: 130ms remaining: 86.4ms

6: learn: 0.5237925 total: 158ms remaining: 67.8ms

7: learn: 0.5092345 total: 185ms remaining: 46.1ms

8: learn: 0.5011426 total: 209ms remaining: 23.3ms

9: learn: 0.4896973 total: 234ms remaining: 0us

0: learn: 0.6550874 total: 13.9ms remaining: 125ms

1: learn: 0.6299542 total: 31ms remaining: 124ms

2: learn: 0.6024134 total: 52.1ms remaining: 122ms

3: learn: 0.5740607 total: 87.7ms remaining: 132ms

4: learn: 0.5553019 total: 109ms remaining: 109ms

5: learn: 0.5338945 total: 154ms remaining: 102ms

6: learn: 0.5164720 total: 189ms remaining: 80.9ms

7: learn: 0.4996422 total: 224ms remaining: 56.1ms

8: learn: 0.4882107 total: 262ms remaining: 29.1ms

9: learn: 0.4793029 total: 278ms remaining: 0us

0: learn: 0.6508561 total: 42.5ms remaining: 383ms

1: learn: 0.6171856 total: 78.3ms remaining: 313ms

2: learn: 0.5877385 total: 114ms remaining: 267ms

3: learn: 0.5612915 total: 152ms remaining: 227ms

4: learn: 0.5409489 total: 177ms remaining: 177ms

5: learn: 0.5249032 total: 202ms remaining: 134ms

6: learn: 0.5148001 total: 211ms remaining: 90.2ms

7: learn: 0.5040501 total: 224ms remaining: 56ms

8: learn: 0.4896151 total: 259ms remaining: 28.8ms

9: learn: 0.4759936 total: 295ms remaining: 0us

>

> print(model)

Catboost

891 samples

7 predictor

2 classes: 'X0', 'X1'

No pre-processing

Resampling: Cross-Validated (3 fold)

Summary of sample sizes: 594, 594, 594

Resampling results across tuning parameters:

depth Accuracy Kappa

4 0.7934905 0.5520102

6 0.7957351 0.5494667

8 0.8080808 0.5791991

Tuning parameter 'learning\_rate' was held constant at a value of 0.1

Tuning

was held constant at a value of 0.1

Tuning parameter 'rsm' was held constant at a value

of 0.95

Tuning parameter 'border\_count' was held constant at a value of 64

Accuracy was used to select the optimal model using the largest value.

The final values used for the model were depth = 8, learning\_rate = 0.1, iterations =

10, l2\_leaf\_reg = 0.1, rsm = 0.95 and border\_count = 64.

>

> # variable importance

> importance <- varImp(model, scale = FALSE)

> print(importance)

custom variable importance

Overall

Sex 67.757

Pclass 17.281

Parch 6.264

Embarked 5.800

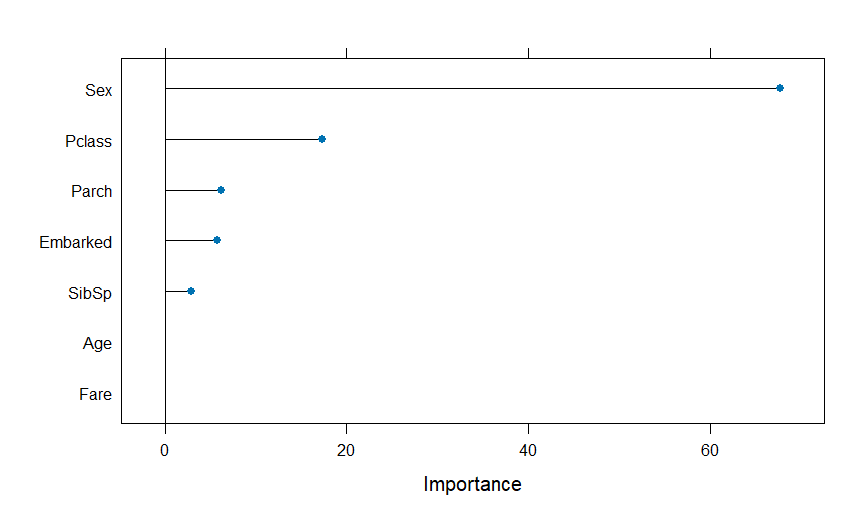
SibSp 2.898

Age 0.000

Fare 0.000

>

> plot(importance)



1. **Kesimpulan**

Model CatBoost yang dilatih pada dataset Titanic menunjukkan bahwa jenis kelamin penumpang adalah fitur yang paling penting dalam memprediksi apakah penumpang tersebut selamat atau tidak. Kelas penumpang juga memberikan kontribusi yang signifikan, sementara fitur lain seperti usia dan tarif tidak berpengaruh signifikan pada model ini. Model ini dioptimalkan menggunakan grid search dan cross-validation untuk memastikan akurasi yang baik dan menghindari overfitting.