The Oscars Award

Data Modelling: Klassifikation

Mathias Petak, Ivo Otero

4 1 2022

Contents

Libraries

```
## Warning: Paket 'sjmisc' wurde unter R Version 4.1.2 erstellt
```

1 Einleitung und Aufgabenstellung

In diesem Teil des Projektes werden verschiedenen Klassifizierungsmethoden beim miteinander vergleicht anhand der Daten aus unserem *Oscar-Dataset*, die für den *ExplorativeAnalysis* Teil bereits verwendet wurde (mit einige Anpassungen, die später erwähnt werden).

Wir benutzen die **Klassifikation**, um eine kategoriale Variable vorhersagen zu können. In diesem Projekt fokussieren wir uns auf 3 unterschiedliche Methoden, die oft in Data Science Anwendungen zu diesem Zweck verwendet werden: *Random Forest*, *Naive Bayes Classifiers* und *Neural Networks*.

Die 3 erwähnte Methoden werden miteinander verglichen, und die Methode mit dem besten Ergebnis beim "Performance Assessment" wird anschließend für die Vorhersage angewendet.

1.1 Was wird vorhergesagt?

Anhand der ausgewählten Methode wird vorhergesagt, ob ein Film nach ihre Features das Oscar-Preis für "Best Picture" (beste Film) gewonnen hat oder nicht. **BONUS:** Anschließend wird auch anhand der Ratings und Golden Globe 2022 Ergebnisse versucht, der Gewinner des 2022 Best Picture Award vorherzusagen (Ergebnis der Auszeichnung erfolgt aber erst im März).

2 Daten einlesen und aufbereiten

Die erste Aufgabe ist die Daten einlesen und für die weitere Verarbeitung aufzubereiten. In diesem Beispiel werden als Erstes die Attributen `Winner` (ob ein Film gewonnen hat oder nicht) und `Category` als `factor` umgestellt.

```
oscars_data <- read.csv(file = '../data/the_oscar_award.csv', header = TRUE, sep = ",", encoding = "UTF-
oscars_tbl <- as_tibble(oscars_data)
oscars_tbl$winner <- as.factor(oscars_tbl$winner)
oscars_tbl$category <- as.factor(oscars_tbl$category)
head(oscars_tbl)
```

```
## # A tibble: 6 x 7
##   year_film year_ceremony ceremony category      name                film  winner
##       <int>         <int>    <int> <fct>         <chr>               <chr> <fct>
## 1      1927          1928        1 ACTOR         Richard Barthelmess The ~ False
## 2      1927          1928        1 ACTOR         Emil Jannings       The ~ True
## 3      1927          1928        1 ACTRESS       Louise Dresser      A Sh~ False
## 4      1927          1928        1 ACTRESS       Janet Gaynor        7th ~ True
## 5      1927          1928        1 ACTRESS       Gloria Swanson      Sadi~ False
## 6      1927          1928        1 ART DIRECTION Rochus Gliese       Sunr~ False
```

2.1 Cleanup

Da bei dieser Arbeit der Fokus bei der "Best Picture" Kategorie liegt, werden wir für die Vorhersagemodelle alle anderen Kategorien aus unserem Datensatz herausnehmen. Da aber das Best Picture Award in der Vergangenheit anders genannt wurde, haben wir diese Kategorien recherchiert und in eine einzelne Kategorie `BEST PICTURE` zusammengesetzt, um später besser damit arbeiten zu können.

```
for (row in 1:nrow(oscars_tbl)) {
  if (str_contains(as.character(oscars_tbl[row, "category"]), "OUTSTANDING")) {
    oscars_tbl[row, "category"] = "BEST PICTURE"
  } else if (str_contains(as.character(oscars_tbl[row, "category"]), "BEST MOTION")) {
    oscars_tbl[row, "category"] = "BEST PICTURE"
  } else if (str_contains(as.character(oscars_tbl[row, "category"]), "UNIQUE AND ARTISTIC")) {
    oscars_tbl[row, "category"] = "BEST PICTURE"
  }
}
```

```
bestMovies = subset(oscars_tbl, category == "BEST PICTURE")
```

2.2 Data Merging

2.2.1 Externe Daten einlesen

Um die Gewinner vorhersagen zu können, müssen wir unseren Dataset mit den Oscar-nominierten Filmen erweitern und neue Variablen hinzufügen, die möglicherweise ein Einfluss auf die Zielvariable "winner" haben. Wir haben uns in diesem Fall für den "oscardata_bestpicture" (*Kaggle: Data on Oscar nominated films between 1960 and 2021*) Datensatz geeinigt, der Daten aus IMDB (Film und Rating-Website) beinhaltet.

```
movie_data_imdb <- read.csv(file = '../data/oscardata_bestpicture.csv', header = TRUE, sep = ",", encodi
movie_data_imdb <- as_tibble(movie_data_imdb)
names(movie_data_imdb)[2] = "film"
head(movie_data_imdb)
```

```
## # A tibble: 6 x 60
##   Category film            Nominee Winner  Year Rating_IMDB Release_date Rating_rtaudien~
##   <chr>    <chr>           <chr>    <int> <int>       <dbl> <chr>                   <int>
## 1 Picture  The Apartment   The Ap~      1  1961         8.3 1960-09-16                 94
## 2 Picture  The Alamo       The Al~      0  1961         6.9 1960-10-24                 64
## 3 Picture  Elmer Gantry    Elmer ~      0  1961         7.9 1960-07                    86
## 4 Picture  Sons and Lovers Sons a~      0  1961         7.3 1960-07-22                 54
## 5 Picture  The Sundowners  The Su~      0  1961         7.2 1961-02-28                 62
## 6 Picture  West Side Story West S~      1  1962         7.6 1961-12-23                 84
## # ... with 52 more variables: Rating_rtcritic <int>, Oscarstat_totalnoms <int>,
## #   Release_Q1 <int>, Release_Q2 <int>, Release_Q3 <int>, Release_Q4 <int>,
## #   Nom_Oscar_bestdirector <int>, Nom_DGA <int>, Nom_BAFTA <int>,
## #   Win_DGA <int>, Win_BAFTA <int>, Nom_GoldenGlobe_bestcomedy <int>,
## #   Nom_GoldenGlobe_bestdrama <int>, Win_GoldenGlobe_bestcomedy <int>,
## #   Win_GoldenGlobe_bestdrama <int>, Genre_action <int>, Genre_biography <int>,
## #   Genre_crime <int>, Genre_comedy <int>, Genre_drama <int>, ...
```

2.2.2 Datensätze zusammenfügen

Für das Zusammenführen der beiden Tabellen wurde die `merge()` Methode genommen, und nur die relevantesten Spalten wurden betrachtet (neu dazugekommen sind "Rating_IMDB", "Win_GolenGlobe_bestdrama", u. A).

```r
merged = merge(bestMovies, movie_data_imdb, by = "film") |>
          subset(select = c(film, ceremony, name, winner, Year, Rating_IMDB, Oscarstat_totalnoms, Nom_Os
```

Weiters haben wir aus Vereinfachungsgründen die "genre" Spalten (die Auskuft geben, ob ein Film einem bestimmten Genre gehört) zusammengeführt, um nur eine Spalte mit jeweiligem Genre des Filmes zu haben.

```r
merged$genre = as.character(NA)

for (row in 1:nrow(merged)) {

  if (merged[[row, 'Genre_action']] == 1) {
    merged[[row, 'genre']] = "Action"

  } else if (merged[[row, 'Genre_biography']] == 1) {
    merged[[row, 'genre']] = "Biography"
  } else if (merged[[row, 'Genre_crime']] == 1) {
    merged[[row, 'genre']] = "Crime"
  } else if (merged[[row, 'Genre_comedy']] == 1) {
    merged[[row, 'genre']] = "Comedy"
  } else if (merged[[row, 'Genre_drama']] == 1) {
    merged[[row, 'genre']] = "Drama"
  } else if (merged[[row, 'Genre_horror']] == 1) {
    merged[[row, 'genre']] = "Horror"
  } else if (merged[[row, 'Genre_fantasy']] == 1) {
    merged[[row, 'genre']] = "Fantasy"
  } else if (merged[[row, 'Genre_sci.fi']] == 1) {
    merged[[row, 'genre']] = "SciFi"
  } else if (merged[[row, 'Genre_mystery']] == 1) {
    merged[[row, 'genre']] = "Mystery"
  } else if (merged[[row, 'Genre_music']] == 1) {
    merged[[row, 'genre']] = "Music"
  } else if (merged[[row, 'Genre_romance']] == 1) {
    merged[[row, 'genre']] = "Romance"
  } else if (merged[[row, 'Genre_history']] == 1) {
    merged[[row, 'genre']] = "History"
  } else if (merged[[row, 'Genre_war']] == 1) {
    merged[[row, 'genre']] = "War"
  } else if (merged[[row, 'Genre_thriller']] == 1) {
    merged[[row, 'genre']] = "Thriller"
  } else if (merged[[row, 'Genre_adventure']] == 1) {
    merged[[row, 'genre']] = "Adventure"
  } else if (merged[[row, 'Genre_filmnoir']] == 1) {
    merged[[row, 'genre']] = "FilmNoir"
  } else if (merged[[row, 'Genre_family']] == 1) {
    merged[[row, 'genre']] = "Family"
  } else if (merged[[row, 'Genre_sport']] == 1) {
    merged[[row, 'genre']] = "Sport"
  } else if (merged[[row, 'Genre_western']] == 1) {
    merged[[row, 'genre']] = "Western"
  }
}

merged = merged |> select(-contains("Genre_"))
```

```
merged$genre = as.factor(merged$genre)
merged$Nom_Oscar_bestdirector = as.factor(merged$Nom_Oscar_bestdirector)

head(merged)
```

```
##                    film ceremony
## 1      12 Years a Slave       86
## 2             127 Hours       83
## 3      A Beautiful Mind       74
## 4    A Clockwork Orange       44
## 5        A Few Good Men       65
## 6 A Man for All Seasons       39
##                                                                                name
## 1 Brad Pitt, Dede Gardner, Jeremy Kleiner, Steve McQueen and Anthony Katagas, Producers
## 2                            Christian Colson, Danny Boyle and John Smithson, Producers
## 3                                          Brian Grazer and Ron Howard, Producers
## 4                                                    Stanley Kubrick, Producer
## 5                              David Brown, Rob Reiner and Andrew Scheinman, Producers
## 6                                                     Fred Zinnemann, Producer
##   winner Year Rating_IMDB Oscarstat_totalnoms Nom_Oscar_bestdirector
## 1   True 2014         8.1                   9                      1
## 2  False 2011         7.6                   5                      0
## 3   True 2002         8.2                   8                      1
## 4  False 1972         8.3                   4                      1
## 5  False 1993         7.6                   4                      0
## 6   True 1967         7.9                   8                      1
##   Rating_rtcritic Nom_GoldenGlobe_bestdrama Win_GoldenGlobe_bestdrama     genre
## 1              96                         1                         1 Biography
## 2              93                         0                         0 Biography
## 3              75                         1                         1 Biography
## 4              90                         1                         0     Crime
## 5              81                         1                         0     Drama
## 6              82                         1                         1 Biography
```

```
sum(is.na(merged))
```

```
## [1] 4
```

```
## Cries and Whispers
merged[74, "genre"] = "Drama"

## M*A*S*H
merged[149, "genre"] = "Comedy"

## Star Wars
merged[212, "genre"] = "SciFi"

## The Godfather Part 2
merged[242, "genre"] = "Crime"

sum(is.na(merged))
```

```
## [1] 0
```

```
# export "merged" für Dashboard visualisierung
write.csv(merged,"../data/oscars_merged.csv", row.names = FALSE)
```

3 Modellierung

In diesem Absatz werden jeweils die Training und Test-Daten erstellt und anschließend benutzt, um die verschiedene Modelle zu trainieren.

3.1 Train und Test-Data erstellen

```
set.seed(23489)
ind = createResample(merged$winner, times = 1)

train = merged[ind$Resample1,]
test = merged[-ind$Resample1,]

nrow(train)
```

```
## [1] 308
```

```
nrow(test)
```

```
## [1] 122
```

3.2 Data Training und Tuning

Für die Vorhersage haben wir uns entschieden, nicht alle Attributen zu nehmen, da es zu einem schlechteren Ergebnis führt (Kappa bei ca. 0). Die beste Attribut-Kombination hat sich ergeben bei der Auswahl von `Oscarstat_totalnoms` (Anzahl an Oscar-Nominierungen für den jeweiligen Film) + `Rating_rtcritic` (Kritikers Rating) + `Win_GoldenGlobe_bestdrama` (boolean, ob der Film die Golden Globes gewonnen hat)

3.2.1 RandomForest

```
model_rf = train(winner ~ Oscarstat_totalnoms + Rating_rtcritic + Win_GoldenGlobe_bestdrama,
                 data = train,
                 method = "rf",
                 preProcess = c("scale", "center"),
                 tuneGrid = data.frame(mtry = 1))
model_rf
```

```
## Random Forest
##
## 308 samples
##   3 predictor
```

```
##   2 classes: 'False', 'True'
##
## Pre-processing: scaled (3), centered (3)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 308, 308, 308, 308, 308, 308, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8337419  0.3622512
##
## Tuning parameter 'mtry' was held constant at a value of 1
```

3.2.2 NaiveBayes

```
model_nb = train(winner ~ Win_GoldenGlobe_bestdrama + Oscarstat_totalnoms + Rating_rtcritic,
                 data = train,
                 method = "nb",
                 preProcess = c("scale", "center")
                 )
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 32
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 54
```

```
model_nb
```

```
## Naive Bayes
##
## 308 samples
##   3 predictor
##   2 classes: 'False', 'True'
##
## Pre-processing: scaled (3), centered (3)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 308, 308, 308, 308, 308, 308, ...
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.7895996  0.3259325
##    TRUE      0.7973665  0.2283357
##
## Tuning parameter 'fL' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 0, usekernel = TRUE and adjust
##  = 1.
```

3.2.3 Neural Networks

```
model_nn
```

```
## Neural Network
##
## 308 samples
##   3 predictor
##   2 classes: 'False', 'True'
##
## Pre-processing: scaled (3), centered (3)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 308, 308, 308, 308, 308, 308, ...
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0.01   0.8018404  0.2814353
##   0.02   0.8043606  0.2769139
##   0.03   0.8065295  0.2889385
##   0.04   0.8060176  0.2787726
##   0.05   0.8073581  0.2856535
##   0.06   0.8099174  0.3111309
##   0.07   0.8106449  0.3139955
##   0.08   0.8109782  0.3148069
##   0.09   0.8110114  0.3113169
##   0.10   0.8106304  0.3105428
##   0.11   0.8117445  0.3135880
##
## Tuning parameter 'size' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were size = 1 and decay = 0.11.
```

4 Performance Assesment und Vergleich

Nachdem wir alle Modelle trainiert haben, werden diese miteinander verglichen, um das beste Modell für die Vorhersage der Daten herauszufinden.

4.1 Vergleich mit PostResamle

```
res = resamples(list(nb = model_nb, rf = model_rf, nn = model_nn))
summary(res)
```

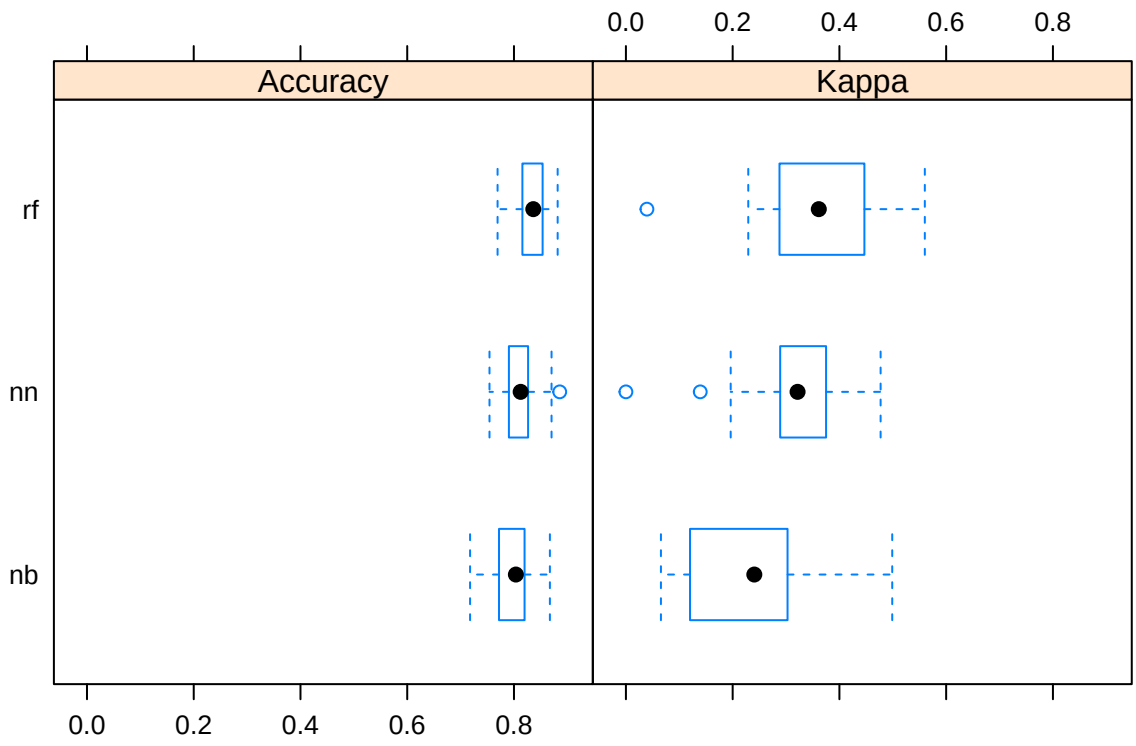
```
##
## Call:
## summary.resamples(object = res)
##
## Models: nb, rf, nn
## Number of resamples: 25
```

```
##
## Accuracy
##         Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## nb 0.7177419 0.7719298 0.8035714 0.7973665 0.8198198 0.8672566    0
## rf 0.7692308 0.8157895 0.8362069 0.8337419 0.8536585 0.8818182    0
## nn 0.7540984 0.7904762 0.8125000 0.8117445 0.8264463 0.8857143    0
##
## Kappa
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## nb 0.06561974 0.1201185 0.2405660 0.2283357 0.3028169 0.4989654    0
## rf 0.03937758 0.2877114 0.3613445 0.3622512 0.4467766 0.5600000    0
## nn 0.00000000 0.2890365 0.3216233 0.3135880 0.3750000 0.4771784    0
```

4.2 Graphische Darstellung

```
## pdf
##   2
```

```
bwplot(res)
```



Auf den ersten Blick kann man gut erkennen, dass der `RandomForest` Modell besser ausschneidet als die zwei weitere Alternativen. Alle Modelle präsentieren eine ähnliche `Accuracy`, aber es gibt ein großer Unterschied beim Vergleich der `Kappa` Kennzahl für jedes Modell.

Da diese Kennzahlen aber nicht die volle Geschichte erklären, werden im folgenden Absatz die `Confusion Matrices` den einzelnen Modellen miteinander verglichen.

4.3 ConfusionMatrix für jedes Modell

4.3.1 Naive-Bayes

```
tab = trunc(confusionMatrix(model_nb)$table)
confusionMatrix(tab, mode = "prec_recall")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction False True
##      False    74   15
##      True      4    4
##
##                Accuracy : 0.8041
##                  95% CI : (0.7111, 0.8778)
##     No Information Rate : 0.8041
##     P-Value [Acc > NIR] : 0.56097
##
##                   Kappa : 0.2039
##
##  Mcnemar's Test P-Value : 0.02178
##
##               Precision : 0.8315
##                  Recall : 0.9487
##                      F1 : 0.8862
##              Prevalence : 0.8041
##          Detection Rate : 0.7629
##    Detection Prevalence : 0.9175
##       Balanced Accuracy : 0.5796
##
##        'Positive' Class : False
##
```

4.3.2 RandomForest

```
tab = trunc(confusionMatrix(model_rf)$table)
confusionMatrix(tab, mode = "prec_recall")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction False True
##      False    76   13
##      True      2    6
##
##                Accuracy : 0.8454
##                  95% CI : (0.7578, 0.9108)
##     No Information Rate : 0.8041
##     P-Value [Acc > NIR] : 0.186577
```

```
##
##                   Kappa : 0.3715
##
##  Mcnemar's Test P-Value : 0.009823
##
##               Precision : 0.8539
##                  Recall : 0.9744
##                      F1 : 0.9102
##              Prevalence : 0.8041
##          Detection Rate : 0.7835
##    Detection Prevalence : 0.9175
##       Balanced Accuracy : 0.6451
##
##        'Positive' Class : False
##
```

4.3.3 Neural Networks

```
tab = trunc(confusionMatrix(model_nn)$table)
confusionMatrix(tab, mode = "prec_recall")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction False True
##      False    74   14
##      True      3    6
##
##                Accuracy : 0.8247
##                  95% CI : (0.7343, 0.8945)
##     No Information Rate : 0.7938
##     P-Value [Acc > NIR] : 0.27028
##
##                   Kappa : 0.3278
##
##  Mcnemar's Test P-Value : 0.01529
##
##               Precision : 0.8409
##                  Recall : 0.9610
##                      F1 : 0.8970
##              Prevalence : 0.7938
##          Detection Rate : 0.7629
##    Detection Prevalence : 0.9072
##       Balanced Accuracy : 0.6305
##
##        'Positive' Class : False
##
```

4.3.4 Interpretation und Entscheidung

Table 1: Vergleich der “Confusion Matrices”

	Naive-Bayes	RandomForest	NeuralNetwork
Accuracy	0.7857	0.8454	0.7959
Recall	0.9231	0.8539	0.9870
Precision	0.8276	0.9744	0.8000
F1	0.8727	0.9102	0.8837

Aus der obigen Tabelle geht hervor, dass das *Naive-Bayes* und das *NeuralNetwork* Modell ähnliche, gute, aber nicht optimale Ergebnisse für die 4 beschriebenen Metriken erzielen. Das einzige Modell, das hiervon abweicht, ist das *RandomForest* Modell, der in allen Kategorien eine sehr gute Leistung aufweist.

Das *RandomForest* Modell hat einen Vorsprung vor den anderen beiden Modellen, da es in allen 4 Metriken um einen kleinen Prozentsatz besser ist als seine Konkurrenten. Aus diesem Grund wäre dieses Modell die beste Wahl für die Vorhersage des Attributs “winner” beim Best Picture Award der Oscar-Verleihung.

5 Vorhersagen mit RandomForest

In diesem Abschnitt werden die Vorhersagen anhand der Testdaten durchgeführt, und ihr Ergebnis wird anschließend ausgegeben.

5.1 Vorhersage der Test-Daten

```
pred_rf = predict(model_rf, test)
postResample(pred_rf, test$winner)
```

```
##  Accuracy     Kappa
## 0.8524590 0.3163138
```

Wie man hier bemerkt, das Modell hat die Oscar-Gewinner mit einer 85,24% Genauigkeit vorhergesagt. Laut dem Kappa-Wert erkennen wir auch, dass diese Vorhersage auch ein Mehrwert gegen einfaches Raten präsentiert.

5.2 Vorhersage der 2022 Oscars

Für die Vorhersage der 2022 Oscars wird in diesem Fall die Nominierungen der "Golden Globes Award" genommen, da die offiziellen Oscar-Nominierungen für "Best Picture" noch nicht bekannt sind.

```
film = c("The Power of The Dog", "Belfast", "CODA", "Dune", "King Richard")
ceremony = c(93, 93, 93, 93, 93)
name = c("Jane Campion, Producer", "Kenneth Branagh, Producer", "Sian Heder, Producer", "Denis Villeneu
#winner = c()
year = c(2021, 2021, 2021, 2021, 2021)
Rating_IMDB = c(7.0, 7.4, 8.1, 8.1, 7.6)
Oscarstat_totalnoms = c(7, 7, 2, 3, 4)
Nom_Oscar_bestdirector = c(NA, NA, NA, NA, NA)
Rating_rtcritic = c(95, 87, 96, 83, 90)
Nom_GoldenGlobe_bestdrama = c(1, 1, 1, 1, 1)
Win_GoldenGlobe_bestdrama = c(1, 0, 0, 0, 0)
genre = c("Drama", "Drama", "Drama", "Drama", "Drama")

goldenglobes = data.frame(film, ceremony, name, year, Rating_IMDB, Oscarstat_totalnoms, Nom_Oscar_bestd
```

```
pred_2022 = predict(model_rf, goldenglobes, type = "prob")
pred_2022
```

```
##   False  True
## 1 0.644 0.356
## 2 0.940 0.060
## 3 0.966 0.034
## 4 0.936 0.064
## 5 0.998 0.002
```

Aus den Ergebnissen geht hervor, dass keine der Filme, die für die Golden Globes nominiert wurden, wahrscheinlich das Oscar für *Best Picture* gewinnen werden. Nichtsdestotrotz, das Film mit den höheren Chancen zu gewinnen ist (nicht überraschend) der Golden-Globe-Gewinner: "The Power of The Dog"!