Group 8
Oscar Award Winning Directors & Actor Bias

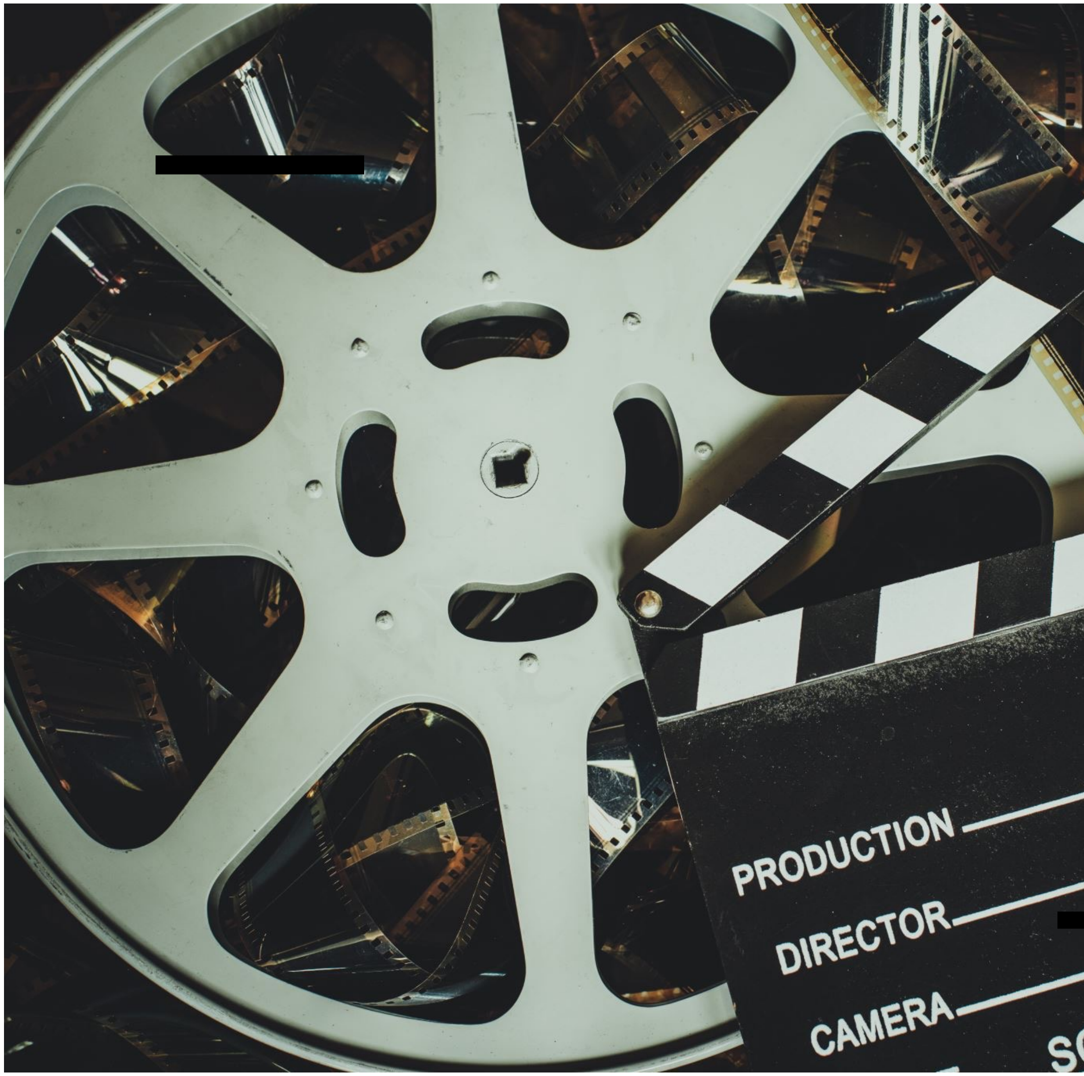
– Carlos Cisneros

– Joe Mota

– Joshua Flores

– Brayan Mendoza

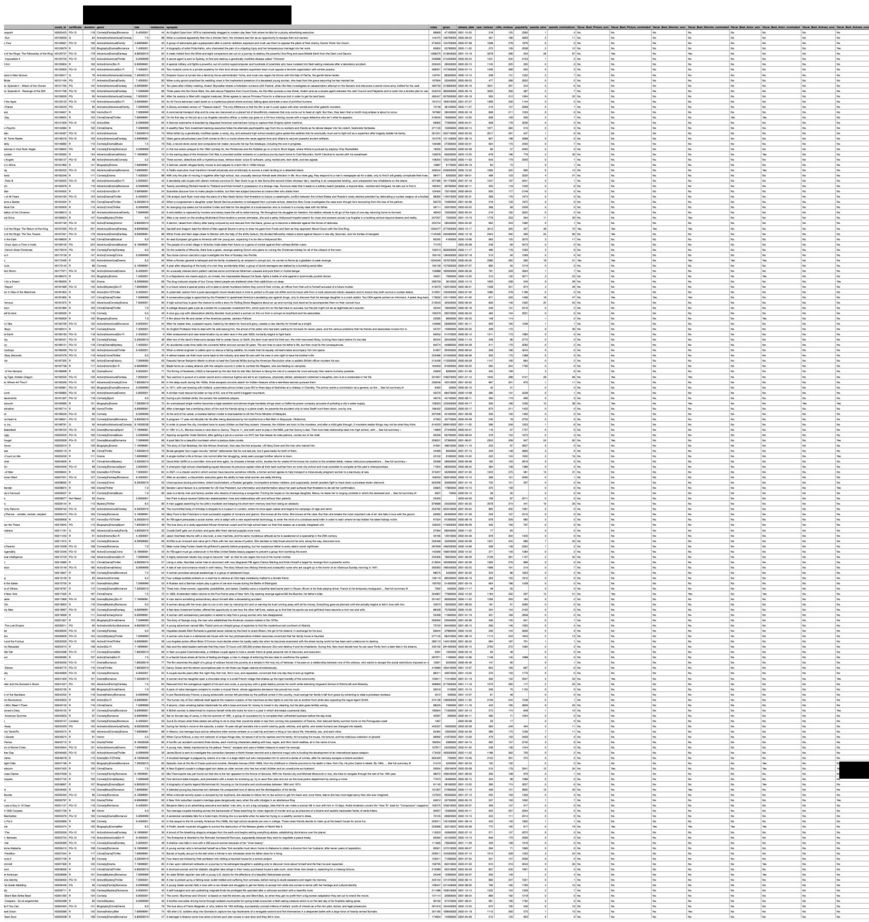
– Alvaro Mendoza



Goal of the Project

- The Goal of the project is to predict Best Director Oscar using characteristics of the Directors movie such as

Certificate, genre, rate, metascore, synopsis, votes, gross, critic_reviews, popularity, awards_wins, Oscar_Best_Director_won, Oscar_Best_Actor_won, Oscar_Best_Actress_won, metascore, awards_nominations, Oscar_Best_Picture_won



Our Data Set/Bias

- Our Data Set demonstrates which movies won Best Director Awards (Oscars) from 2000-2017
- We chose this data as we believed it would be great for predicting winners using non-bias characteristics of the movie.
- The winners arent listed with any identification(race, gender, sexuality or age)

Cleaning Data

– For the data cleaning process, we dropped all the unnecessary columns in the dataset as it contained 119 columns, which we only needed 16.

– We kept the ones that would help us with the project as it contained valuable data that would facilitate t he prediction process such as metascore, votes, gross income, whether it won best director, actor, picture, etc.

```
[144] # TODO:: Use the shape member variable to observe the shape of our datase
      df.shape
      #Remove
      df.columns

      import pandas as pd

      # drop the column you want to delete
      dontDrop = ['certificate','genre','rate','metascore','synopsis','votes','
      for i in df.columns:
          if i not in dontDrop:
              df = df.drop(i , axis=1)
          elif i in dontDrop:
              print(i)
      df.columns

      # write the updated dataframe to a new CSV file
      df.to_csv('Big_oscar.csv', index=False)
```

```
[144] certificate
      genre
      rate
      metascore
      synopsis
      votes
      gross
      critic_reviews
      popularity
      awards_wins
      awards_nominations
      Oscar_Best_Picture_won
      Oscar_Best_Picture_nominated
      Oscar_Best_Director_won
      Oscar_Best_Actor_won
      Oscar_Best_Actress_won
```

Cleaning Data Continued

– Another useful process that helped with both the readability and coding segment converting data of Object type that was represented with a 'No' and 'Yes', such as Oscar_Best_Director_won, and Oscar_Best_Actor_won, among others, which represented if the movie had won Best Director/Actor to 0 for 'No' and 1 for 'Yes'.

```
oscarWins = ['Oscar_Best_Picture_won','Oscar_Best_Director_won','Oscar_Be
for i in oscarWins:
    df[i] = df[i].replace({'Yes': 1, 'No': 0})

df['rate'] = df['rate'] * 10
```

Data Analysis Scatterplot on 'Meta score' and 'Rate'

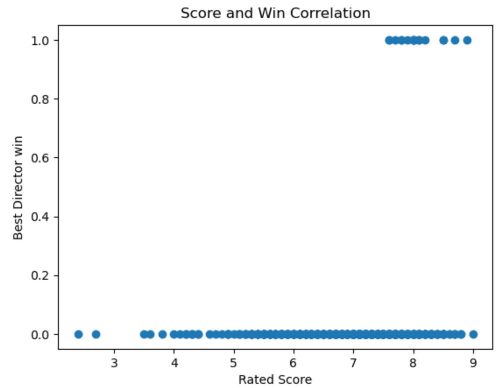
```
[1289] x = df['Oscar_Best_Director_won']  # assuming this line is already defined
# TODO::Set Y variable
y = df['metascore']
# TODO::Generate a new scatter plot for a different pair of variables
plt.scatter(y, x)
plt.ylabel('metascore')
plt.xlabel('Best Director win')
plt.title('Meta score and Win Correlation')
plt.show()
```

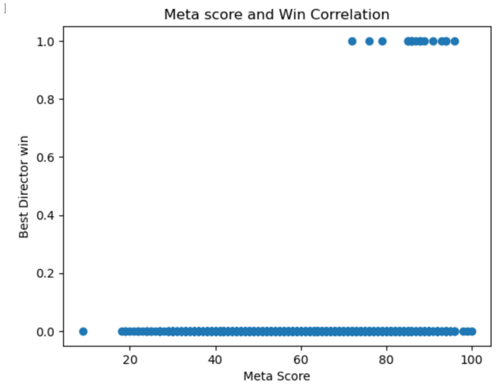
```
x = df['Oscar_Best_Director_won']  # assuming this line is already defined

# TODO::Set Y variable
y = df['rate']
# TODO::Generate a new scatter plot for a different pair of variables
plt.scatter(y, x)
plt.ylabel('Best Director Win')
plt.xlabel('Rated Score')
plt.title('Score and Win Correlation')
plt.show()
```

In our effort to conduct a comprehensive analysis of a sizable dataset, we aimed to obtain a nuanced understanding of the key factors contributing to the attainment of an Academy Award. Through our research, we discovered that two crucial variables - namely, the meta score and rate - appear to strongly influence the likelihood of winning an Oscar for Best Director. To visually represent this relationship, we plotted these two variables on the x-axis of two separate graphs and compared their values against the occurrence of Best Director wins on the y-axis. This was achieved by generating a scatter plot through the implementation of the code above.

Data/analysis Scatterplot

– We developed two scatter plots, one that displayed Oscar best director won with rate (rating score) and Oscar best director won with meta score (critic rating). We saw a significant correlation between in both scatter plots and what we interpreted from these two plots is that the higher the rating/meta score is that the higher rating/meta score the more it influenced the chances of winning. However, the was a low standard deviation and the cluster formed around 7 to 8 for our rated score/best director win scatter plot and 80 to 100 for our meta score/best director win scatter plot. After evaluating this data, the meta score has more of an influence on best actor winning, but the rating didn't necessary have as much of an influence.





Gross Income And Winners/Losers Data Analysis

- With this data we can show the average of the gross income of all the movies that won is $125,722,142.17
- Compare this to the data of the average gross income of the all the movies that didn't win is $89,186,387.56
- Showing that if your gross income is higher than $100,000,000 you will have a <0% chance of winning Best Director Oscar

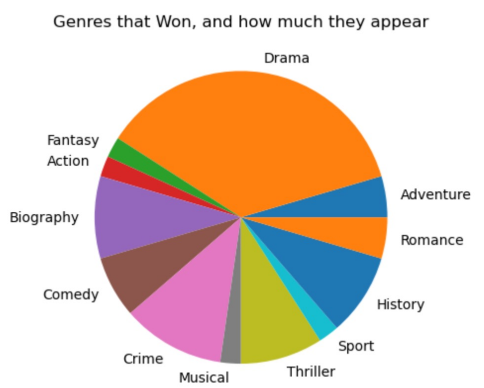
Filtered table for 'Oscar_Best_Director_won' 'YES'

	Oscar_Best_Director_won	gross	Average
			125722142.17
	Yes	377020000	
	Yes	124110000	
	Yes	32519322	
	Yes	170710000	
	Yes	83030000	
	Yes	100420000	
	Yes	132370000	
	Yes	124980000	
	Yes	74270000	
	Yes	15700000	
	Yes	141320000	
	Yes	274080000	
	Yes	138800000	
	Yes	44667095	
	Yes	183640000	
	Yes	42340000	
	Yes	77300000	

	'gross' for 'Oscar_Best_Director_won' 'No'		
Over a 1000 rows of losers		Average is 89186387.56	
		Oscar_Best_Director_won	No
		gross	
		936630000	
		760510000	
		652180000	
		623280000	
		610800000	
		533320000	
		504010000	
		501900000	
		458990000	
		448130000	
		436470000	
		424650000	
		423030000	
		414980000	
		408990000	
		408080000	
		408000000	

Genre Data Analysis/ with Plot

[48]



[62]

```
dict = {}
genresWon = []
for index, item in enumerate(df['genre']):
    genre_list = item.split('|')
    for j in genre_list:
        if j not in dict:
            dict[j] = 1
        else:
            dict[j] += 1
    if(df['Oscar_Best_Picture_won'][index] == 'Yes'):
        genresWon.append(genre_list)

winDict = {}
for i in genresWon:
    for j in i:
        if j in winDict:
            winDict[j] += 1
        else:
            winDict[j] = 1

genreNameWin = list(winDict.keys())
genreCountWin = list(winDict.values())

plt.title("Genres that Won, and how much they're seen in the win")
plt.pie(genreCountWin, labels = genreNameWin)
plt.show()
```

- Looking at the pie chart, we can separate winning titles into 4 chunks, being Drama, Bio, Crime, and History.
- The chances of winning an Oscar is significantly higher, if the movie genre is based of those 4 categories

Machine Learning Model (Supervised)

We trained our machine learning model on data such as the number of previous awards the movie had won, the number of awards it was nominated for, the metascore (weighted average of top critic reviews), the average movie rating, and the gross amount of money it made. All our data was converted to numbers.

The main issue we faced was that our dataset consisted only of specific Oscar wins from 2000-2017. This meant that we had only 17 wins to train our data on, while we had another 1,077 movies in our dataset to train on. We needed more winning examples. To address this issue, we tried adding a few random duplicates of the winning data while also reducing the losing data. This led to better outcomes. On the left, we have an example of training the data on 51 random nominated movies, where 17 won best director in this case.

[redacted]

```
notWon = df[df['Oscar_Best_Director_won'] == 0]

numDropRow = int(len(notWon)//1.03)
rtd = notWon.sample( numDropRow , random_state=42).index
df.drop(rtd, inplace=True)
```

```
print(len(df['Oscar_Best_Director_won']))
X = df[[ 'rate','metascore', 'awards_wins', 'awards_nominations','critic_reviews','votes','gross']]
y = df['Oscar_Best_Director_won']
df.fillna(df.mean(), inplace=True)

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.3, random_state=42)
# Create and train the model
model = RandomForestClassifier()
model.fit(X_train, y_train)
didWin = df[df['Oscar_Best_Director_won'] == 1]
print(len(didWin))
# Make predictions on the testing set
y_pred = model.predict(X_test)

# Evaluate model performance
accuracy = accuracy_score(y_test, y_pred)
print("Model accuracy:", accuracy)
print("Predicted values:", y_pred)
print("   Actual values:", y_test.values)
```

```
51
17
Model accuracy: 0.8125
Predicted values: [1 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0]
   Actual values: [1 0 0 0 1 0 0 0 0 0 1 1 0 1 0 0]
```

Conclusion (What we learned)

- (Data Cleaning)We learned that big data sets like ours need to be cleaned and how it is important to compare same data types to make things easier.
- (Analyzing)We learned how to analyze data and see which characteristics affect the probability of the winner.
- (Machine Learning Model)We learned how to manipulate the model and train on to create greater accuracy on the tested dataset using (Supervised Learning).

Citations

DATASET (BELOW)

- **Oscars nominated Movies 2000-2017(Movies) 119 columns:** https://www.kaggle.com/datasets/vipulgote4/oscars-nominated-movies-from-2000-to-2017

ARTICALS (BELOW)

- **How does a film win an Oscar?:** https://www.euronews.com/2019/02/21/how-does-a-film-win-an-oscar
- **How does Oscar voting work?:** https://variety.com/feature/who-votes-on-oscars-academy-awards-how-voting-works-1203490944/
- **How to win an Oscar?:** https://www.vogue.in/culture-and-living/content/how-to-win-an-oscar-award


Any
Questions???