08_MovieMining_Part4

Discovering Trends in Film Runtime

Caleb Stillman
Computer Science Post Bacc.
University of Colorado Boulder
Boulder Colorado United States
cast9172@colorado.edu

Graydon Sinclair
Computer Science Post Bacc.
University of Colorado Boulder
Boulder Colorado United States
grsi7978@colorado.edu

Stephen Tynan
Computer Science Post Bacc.
University of Colorado Boulder
Boulder Colorado United States
stty1424@colorado.edu

Abstract

The goal of this data mining project is to discover trends in film runtime such that the findings can be applied in the film industry to produce more successful movies. The sample dataset contains information on over 950,000 films. The dataset will be mined for information, specifically to look at the runtime feature's impact on the other features, and vice versa. Among the goals in determining what relationships exist with the runtime are to identify the ideal film length associated with high ratings. Some success was achieved with certain models in furthering this goal - for instance, the Bayesian Belief Network did show that, generally speaking, longer films correlate to better ratings, although there is a threshold beyond which that ceases to be true.

Introduction

Group 8's project goal is the mining of data from a repository with information on over 950,000 films. The mining will have a focus primarily on the runtime feature of the data. It will look at both its impact on other features and the way other features impact it.

Some questions that will be answered via data examination include whether it is possible to predict the runtime of a film based on its genre, country of origin, and/or date of release. Another goal is to identify the ideal film length associated with high ratings.

The intent of determining if these relationships exist, and to what extent, is that this information could be applied within the film industry. This application would theoretically help produce movies with a greater likelihood of positive public reception, resulting in an increased box office.

Related Work

A survey of existing literature suggests that some work has been done to examine the impact of runtime on other facets of films. However, the work has been minimal.

A piece in Cognitive Research Journal [1] speaks to runtime tangentially as it relates to trends in the pacing of films throughout history. However, the author examines a limited 210 films and only includes films within an arbitrary runtime cutoff at the two-and-a-half-hour mark.

Another article from Film and Digital Media [2] does focus on runtime but is a shallow piece. The intent of this project is to mine further information than is presented in this strictly topical article.

IMDBPro [3] published an article that focuses on how the success of a film is measured, which is related to this project, but it does not speak to runtime at all. This may lead one to believe that runtime is an overlooked aspect of success.

Data Set

The data set is sourced from the website/app Letterboxd and hosted on Kaggle [4]. Letterboxd is a social media film site that aggregates film data and viewer interactions such as ratings, likes, and views with said films. The data set has been downloaded locally in case of an unforeseen event where it is no longer available via the web.

The data set is quite large. It includes 10 sub-sets of data and has information on over 950,000 films. Each of these films can have many entries across the subsets of data. For instance, one film may have 3 genres (rows) of data in the genre sub-set alone. The sun-sets include information on genres, country of origin, actors, crew, studios, languages, poster urls, release type and date information, themes, and the movies themselves. The sub-sets vary in size. As mentioned above each film can have many rows in each sub-set other than the movies sub-set. No sub-set has more than seven columns (including the primary key).

The data set is updated annually. It was last updated July 2024, meaning the runtime data included is very up to date for current research. However, when looking at some features, such as ratings, it may be important to note that this feature may not have stabilized for a movie that came out close to the data set update. This is another candidate for domain knowledge-based outliers for certain features.

It is important to note that most of the data in these data sets is categorical. Therefore, if regression is to be done on any of these subsets or the final monolithic set then some encoding will be required.

Main Techniques Applied

This project began with EDA, including concurrent data cleaning and preprocessing.

As part of the EDA process outliers in the data were identified and handled. For instance, a cursory glance at the data showed that there were movies over the 7-hour mark in the dataset. Through knowledge of the area (area expertise) and research it was determined that these outliers were not generally bad data but in fact actual movie lengths. The extreme outliers were removed to not impact the data as these singular movies are not common and are often experimental films that should not be considered when attempting to look at the ideal runtime for the average studio or viewer. Additionally, the data set was last updated in July 2024, meaning that any films with release dates past June 2024 likely had bad or limited ratings data as only select viewers would have seen it. For the purposes of this research these films were also removed from the dataset.

Not only were the outliers removed but the overall data characteristics were recorded and documented. During EDA, visualization was leveraged to help determine if any patterns arose. Visualizing the data and documenting characteristics helped understand the distributions and relationships between features and influenced which features were kept for which parts of the project.

In an earlier iteration of this project, it was discussed that other datasets may be brought in to get additional information on box office, budget, and ratings for films. Since the initial planning it had been decided to abstain from these additions. The primary dataset from Letterboxd includes viewer ratings which can be indexed as a success metric. It was determined that there is already a large sum of literature on the analysis of the box office and budgets of films [3] and additional research would be less impactful than that pertaining to runtime. As there are no other datasets being leveraged there was less data preprocessing and integration than initially planned.

Some integration was still required however, as the Letterboxd data set is split into ten separate subsets in CSV format. A primary key is shared across all these subsets allowing for merging during data frame creation. Other steps in the preprocessing included transforming categorical data into numerical format, using one-hot encoding, so that the information could be more easily parsed. During the evaluation process of the data, it was determined that no additional classification was required, e.g., the assignment of ratings to categories such as great, good, okay, poor, and bad.

The evaluation stage came after the EDA, cleaning, and preprocessing, wherein primary features and correlations were discovered. Mean Squared Error (MSE), regression analysis, Mean Absolute Error (MAE), precision, recall, F1 scores, and lift were just some of the measurements that were used to determine the strength of correlation and general relationships between the features mined.

For evaluation, accuracy measurements were used when looking at the success of classification prediction. Lift was measured to look at the performance and importance of the associations found. MSE was used for regression analysis such as the prediction of runtimes. Precision and recall were used to measure the success of model predictions. F1 scores were also used for evaluation of model success. Matplotlib and Seaborn were leveraged to create visual plots to aid in evaluation, such as providing a comparison between predicted and known values.

Python, and its associated libraries were the primary tools for this project. The following associated libraries were used for mining the data; Pandas for producing and manipulating data frames, NumPy to aid in calculations, Seaborn and Matplotlib to produce plots and other visualizations, Pgmpy for building a Bayesian network, and Scikit-learn to help build regression models. There was initial discussion around the use of Orange, but due to the size of the data set it was determined to not be ideal for this project.

Milestones

Milestones Met :

1.1 Identification

Completion date: 10/01/24

1. Identification of viable datasets
2. Identification of the primary and secondary features.

2.1 EDA – Cleaning

Completion date: 10/15/24

Cleaning of data:

Removal of NaN and bad data

Identification and potential removal of outliers (by both domain expertise and statistical determination)

2.2 Data Preprocessing

Completion date: 10/31/24

Data has been combined into a singular set containing all pertinent data.

Transformation of data – e.g. one-hot encoding

Trend analysis and precursory evaluations

3.1 Analysis

Completion date: 11/30/24

Complete research and analysis on the data. Apply evaluation methods and add visualizations.

Build a predictive model and compare model to test data.

Apply methodologies to the following big questions (more may arise during EDA):

1. What is impact of runtime on rating? / What is the ideal runtime?
2. What is impact of genre on runtime?
3. What is impact of country of origin on runtime?
4. What is the impact of runtime on rating by country of origin?
5. What is the impact of runtime on rating by genre?

4.1 Finalize Results

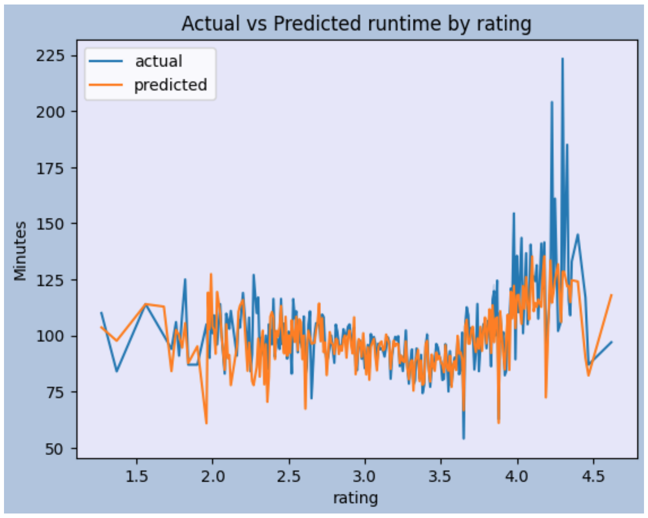
Completion date: 12/5/24

1. Finalize evaluation
2. Synthesize results – determine if there are new questions to be asked
3. Produce final presentation.

Key Results

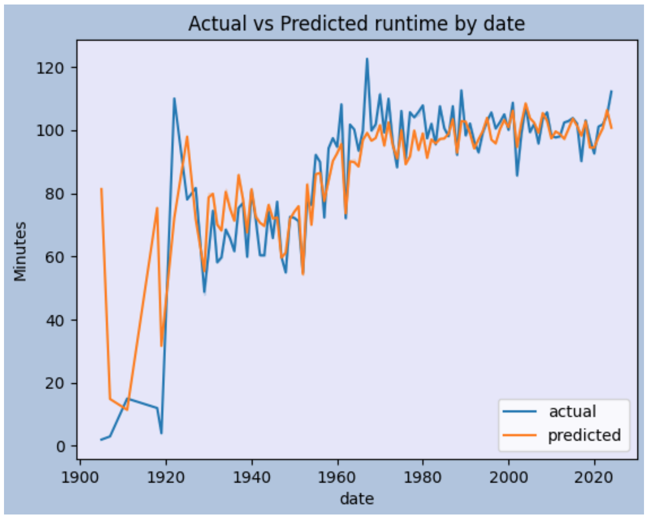
As mentioned in the Main Techniques Applied section above, visualizations were leveraged during the preliminary EDA to determine whether any patterns arose in the data, both those expected and surprising. To this end, the data was fitted and a trained linear regression model on the "minutes" (runtime) feature was constructed. This was used to plot a few graphs: Actual vs. Predicted Runtime by Rating, Actual vs. Predicted Runtime by Date, Actual vs. Predicted Linear, and Rating vs. Minutes. These are each addressed in turn below.

The Actual vs. Predicted Runtime by Rating graph served well as a test of the preliminary linear regression model.



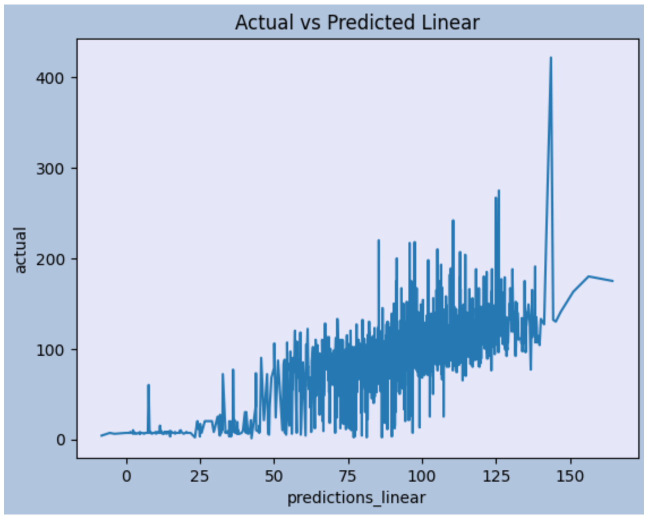
The results of the graph showed an encouraging level of correlation between the actual and the predicted runtimes by rating, particularly among middle-range values, indicating that one of the presumed most important questions was indeed worth further investigation. That said, the graph revealed that the model could use tuning to provide better predictions around the lower and upper ranges of ratings.

Another feature that was graphed with runtime was the release date. The interest in a preliminary graph of these features together at such an early stage was to primarily discern how trends in runtime have developed over time.



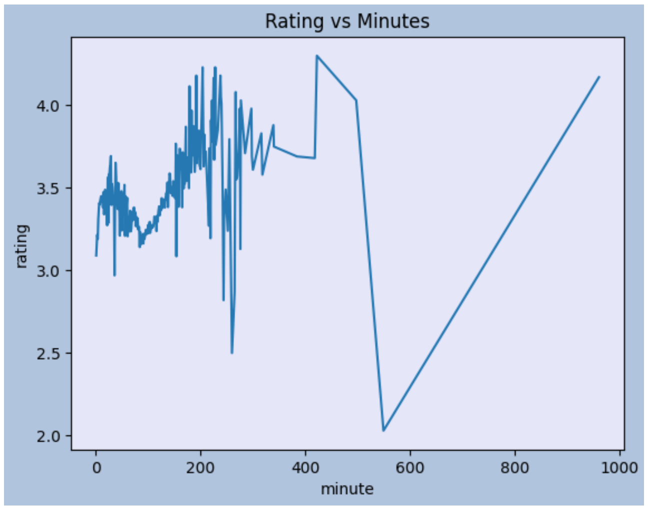
The results of this graph did largely confirm expectations, showing that early films had a much shorter average runtime. There was a large spike in the average length of films in the early 1920’s, stabilizing at a much higher range of runtime (60-80 minutes, up from <20 minutes) beginning around the release of the first non-silent film, The Jazz Singer, in 1927. Once sound was introduced, runtimes stayed in roughly the 60–80-minute duration range for about the next 20 years. Starting in the 50’s, film length began to creep up over the course of the next decade, stabilizing again in the late 60’s/early 70’s to an average length range between 90 and 110 minutes, a trend in film length which has continued throughout the present day. Here again, the linear regression model showed that the model predictions followed actual values to an encouraging degree (the exception being in the earliest two decades of film history).

In speaking about the model, the next plot was of actual runtime values vs. predicted linear regression values. This was used to determine the initial efficacy of the linear regression model.



One thing of note in this graph is that one can see that there are not many films included in the dataset with runtimes between 0 and 25 minutes, and there are a few included in the dataset with runtimes between 25-40 minutes, but after this point is where the data really spikes. This was as expected, as the data set was meant to capture feature films. The Academy of Motion Picture Arts and Sciences, the American Film Institute, and the British Film Institute all consider feature films to be those with runtimes greater than 40 minutes [5]. The data is most tightly clustered around the 45–138-minute range, which is also to be expected, as this is reflected in the previous graph as well. However, this graph also showed that fine-tuning the performance of the linear regression model was necessary. It was also discussed that RMSE, or other metrics may be used to supplement the findings of the linear regression model.

Lastly, a graph of the pure rating vs. minutes of runtime was created, without also graphing the results of the model.



The most striking revelation of this plot was that further cleaning of outliers was necessary, as there was not much data worth noting past the 350 minute mark, but there was data in the 400-1000 minute range which was probably contributing to the skewing of the linear regression model.

All told, the results thus far had been largely encouraging, seeing as the data backed up expected conclusions about known runtime trends, and that the model provided was at least a good start on a prediction model for answering some of the proposed questions.

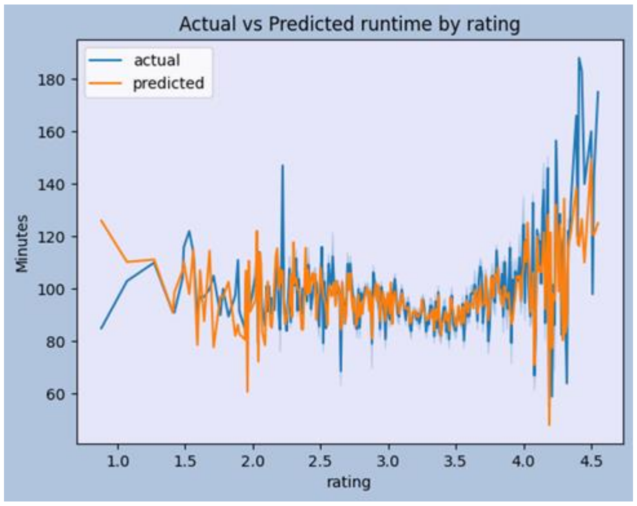
Linear Regression Model

As can be seen, the linear model does a pretty good job of discerning the runtime based on factors such as language, studio name, rating, release date, etc. By visualizing the data, it became apparent that there were some global outliers that may have been throwing off the data. A more effective model may be constructed if these outliers were ignored, as it isn't expected that the model would be particularly good at predicting a global outlier anyways. The current MSE was 570.4679, and the MAE was 15.3628.

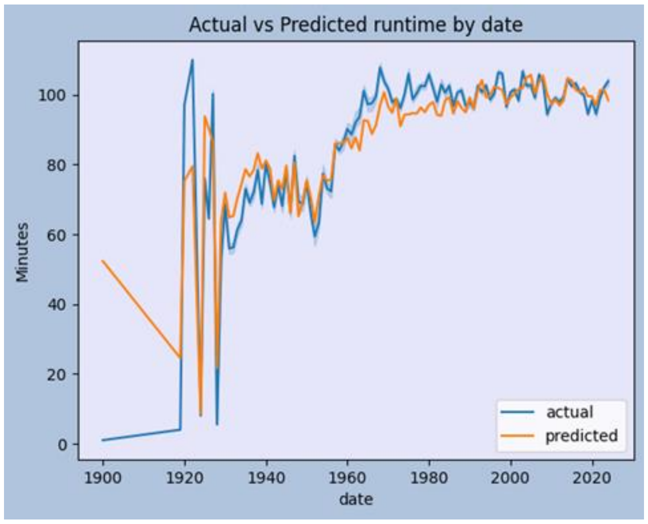
These are strong values for a simple linear regression but there was ample room for improvement.

The first step in improving the model was to rid it of any global outliers. In order to do this the z.score module from "SciPy.Stats" was run on the minute column to remove any outliers within 3 standard deviations of the mean. This proved effective, reducing both the MSE to 426.0888, and the MAE to 14.8819.

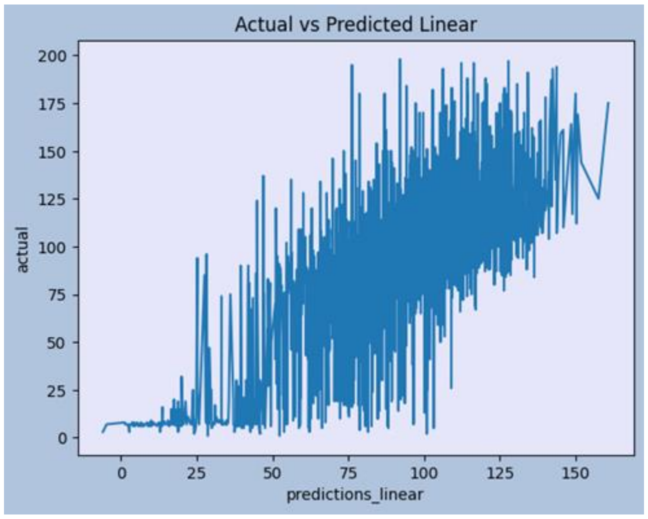
As the goal is not to attempt prediction for outliers the new, improved model appeared to be more effective at predicting for the majority. The model could potentially be further improved by limiting data to only 2 standard deviations but after some deliberation it was determined that movies are often over 2 hours these days so these additional limitations to the model, using historical data that is mainly shorter in length, would skew the results too much. In future modelling it would be interesting to break the data up into time periods to see if it could be more effectively modeled. For example, bucketing the data into 5–10-year periods before modelling may yield additional insights.



With the updated model the fit improved from 1.5 to just below 4.5. However, the model still struggled to predict the lower end of ratings which could be the work of outliers left in the data and would require more analysis.



As can be seen when the data is visualized there were collective outliers still present that may have been throwing the data off. Inspecting the beginning of the "Actual vs Predicted runtime by date" graph it is apparent that the model is particularly ineffectual for the years 1900 - approx. 1930. The model could potentially improve further with the removal of those outliers.

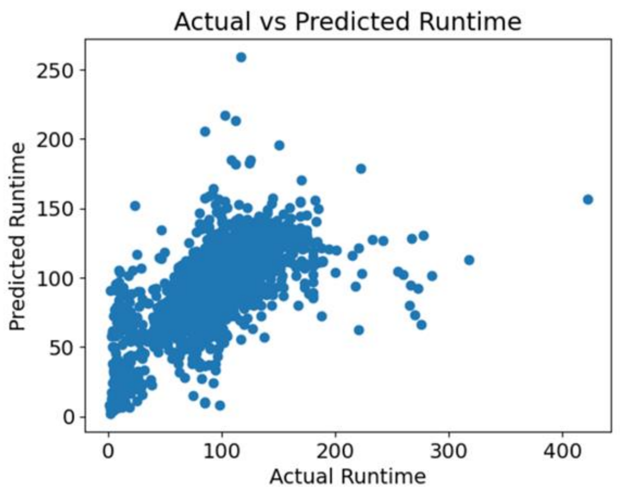


When inspecting the Actual vs Predicted graph, the model did a much better job in this iteration as the distribution is much closer to the y=x line that would be expected if the model were 100% effective. It still struggled a little bit with the upper and lower bounds and was more likely to under than over predict based on the distribution of data points.

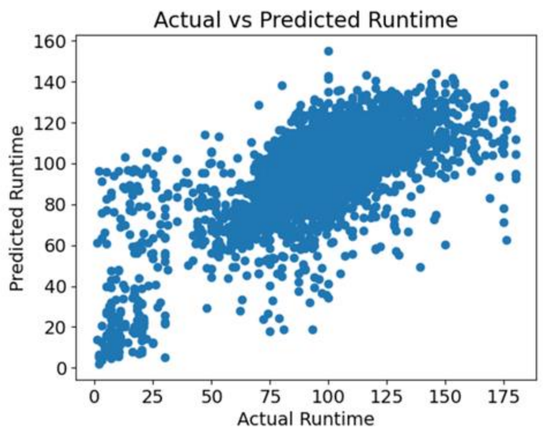
Since the model can effectively predict runtime based on independent variables such as date, studio, release country, language, etc. it can be concluded that there is correlation between the two. Additional tuning was attempted to improve the model by dropping any variables that were not providing a strong correlation. However, this proved ineffectual as it caused a slight increase in MSE and MAE as a result even after removing outliers. For this reason, all variables were included, and the focus was on removal of outliers to improve the model.

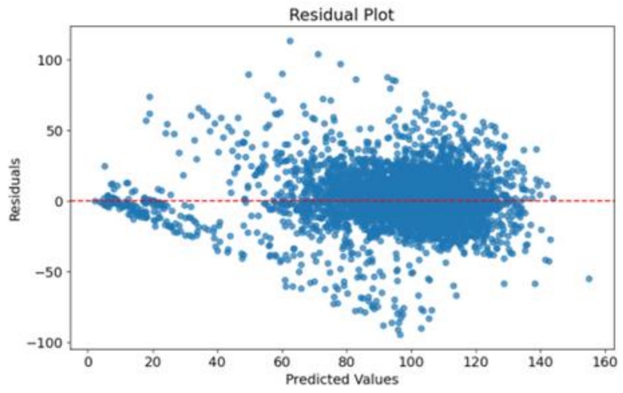
Random Forest Regression Model

To see if it is possible to predict runtime based on other features a Random Forest Regression model was also constructed. This model type was chosen for its efficiency in dealing with large data sets with many features, stability regarding outliers, resistance to overfitting, and its feature importance capabilities which are used in the tuning of the model. The model was built off the refined data set that had gone through preliminary EDA. The target of the model was set to 'minute' (runtime feature) and training and test data were created, with a 70/30 split respectively. The model was trained on the data and then evaluated. This model produced an MSE of 446.15 and a mean R-Squared value of 0.615 across five cross-validation calculations. With a middling R-Squared score, 62% of the variability in the 'minute' target could be explained by the other features included in the model. This score could certainly be better but is not terrible. The MSE of 446.15 indicated that the average squared difference between the actual and predicted values was 446.15. This means that the average prediction was about 21 minutes off from the actual runtime. This may seem small but in relation to movies that only run for a few hours on average this was a sizable discrepancy. Looking at the scatterplot of the Predicted Runtimes vs Actual Runtimes it was clear that there were some outliers but a general positive correlation:



To test and improve the model entirely new models were built on trimmed data. Initially the feature importance was determined and any feature with an importance less than 0.0001 was dropped to help with the sparsity of the data. This dropped 64 feature columns from the data set. 184 columns remained. The model with this new smaller data set produced an MSE of 427.51 and a mean R-Squared score of 0.591. A noticeable improvement in the MSE and a slight decrease in the R-Squared score. Looking at the graph above and the outliers it was noticed that most of them fell beyond the 200-tick mark. To further improve the prediction capabilities of the model all movies longer than 3 hours were dropped. This arbitrary length was decided upon due to the average length of movies being ~141 minutes, well below the 180 mark. These changes yielded an MSE of 309.124 and a mean R-Squared value of 0.7063. Below is the Actual vs Predicted values plotted, and the Residuals of the Predicted Values plotted. Based on the increase in the residuals as the predicted values increase it can be assumed that variance of the errors in the model was not constant. This was not ideal, but the general scattering of the residuals around the horizontal axis indicates a balanced model. Also worth noting, there are clearly some outliers in the graph indicating some data did not fit the model well.





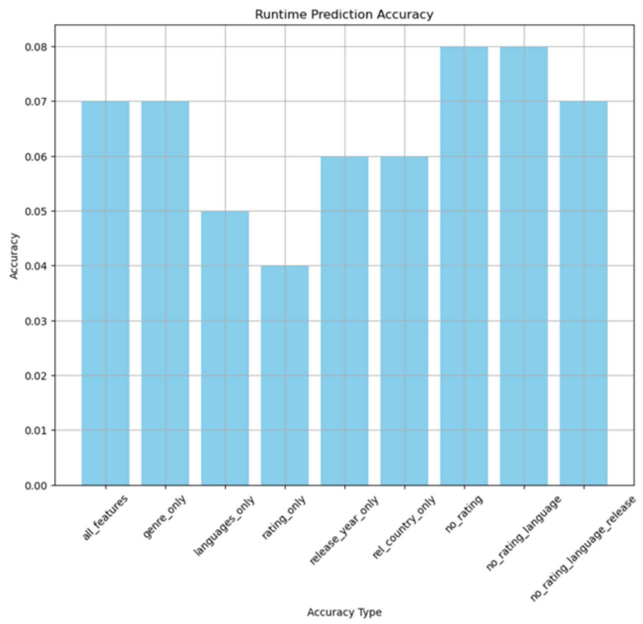
Other models were built such as one using just genre information as the features, the same was done again with countries of production, release dates, and ratings of the movies. All yielded poorer results than the pruned data described in the paragraph above. This indicates that the model is best at predicting movies under 3 hours long but with only data from some studios, release countries, countries of origin, languages, and all the genres and release dates.

SVM

Like the Random Forest Regression attempts at predicting runtime, a Support Vector Machine (SVM) was built to see if it had better performance than the Random Forest. The initial linear SVM was built using all the features in the data set that went through the preliminary EDA. This led to a very low accuracy score of only ~6.5%. The R-Squared value was also extremely low at -0.119, while the MSE value was very high at 1335.45. These are much lower and higher respectively than the same scores for the Random Forest Regressor.

Through hyperparameter tuning it was determined that an SVM with a Radial Basis Function (rbf) kernel would perform better than the linear. This change only caused a minimal improvement in accuracy, but a huge improvement in R-Squared and MSE. The accuracy gained 1%, while the R-Squared became 0.4496 and the MSE dropped to 656.61. However, these scores are still worse than those of the Random Forest Regressor Model.

Further testing the SVM using specific features to predict runtime, such as genres, languages, ratings, release date and country was done. All these models performed worse than the rbf-based SVM mentioned in the previous paragraph with the exception of a single one. The model that performed better than the others was built without the rating or language features. This model produced an accuracy of only 8% but an MSE of 661.17 and an R-Squared value of 0.4797. This was the best performing model of all the iterations. Even with this improved feature selection and hyperparameter tuning the SVM model did not perform nearly as well as the Random Forest Regression Model. Below you can see the limited accuracy of some of the different iterations of the SVM model.



Bayesian Belief Network

The Bayesian Belief Network was chosen as one of the models to explore for this dataset because its representation of conditional probabilities between variables is useful given that the primary application is to allow filmmakers to predict which variables will make their movies most successful. As such, the raw data given by this model type was some of the most easily understood and applied towards this goal. Initially, an attempt was made to automate the generation of the BBN's model structure using the hill climb search algorithm. Hill climb search is a greedy search algorithm that can often be successful in learning the optimal structure of BBNs. However, this was where the drawbacks of a network model became apparent in the form of hardware limitations. A hill-climb-generated structure could not be completed, requiring vastly more space and memory than was available.

To address this issue, smaller data frames were made by dropping certain column groupings, with the resulting data frames built with an eye towards answering questions about a shared feature such as genre or studio, while omitting most other columns. It should be noted that these data frames still included the "minute" feature, as runtime was the primary concern in relation to all other features. This strategy was successful in allowing the hill climb search algorithm to produce BBN structures, but unsuccessful in that hill climb search proved ineffective at finding the optimal structure for any of them. This was likely due to the one-hot encoding scheme that was chosen during earlier stages of data cleaning and preprocessing - the chosen encoding was picked with an eye specifically toward linear regression models and functioned poorly for building a network model structure.

At this juncture, trying to automate the task of defining the structure of the BBN was abandoned. The next approach was to try and build smaller BBNs with the same feature groupings as in the last hill climb search attempt, but to manually define their structure such that the resulting edges would produce the desired data for analysis. However, hardware limitations again proved insurmountable with these attempts becoming too large and terminating in Memory Errors.

The only avenue left to produce a usable Bayesian Belief Network was to manually define very small network structures that involved only two or three features apiece, despite the obvious drawbacks involved. Ideally for this model type, one BBN would be produced which encapsulated all relevant target features to query - runtime, release date, rating, genre, studio, release country, release type, language, and so forth. The concern about having to produce such small BBNs is that many models would need to be created which could easily become time consuming and hard to track. Additionally, certain relationships might be lost, or skew might be introduced, particularly in the cases where the encoding scheme generated many feature columns out of one feature (i.e., the studio column in the original dataset became 73 features, one column for each unique studio value). With the hardware limitations imposing such strict limitations on the number of features that can produce a BBN at all, the question is how can the conditional probabilities of runtime in relationship to studios be accurately modeled if they're broken out across many small models?

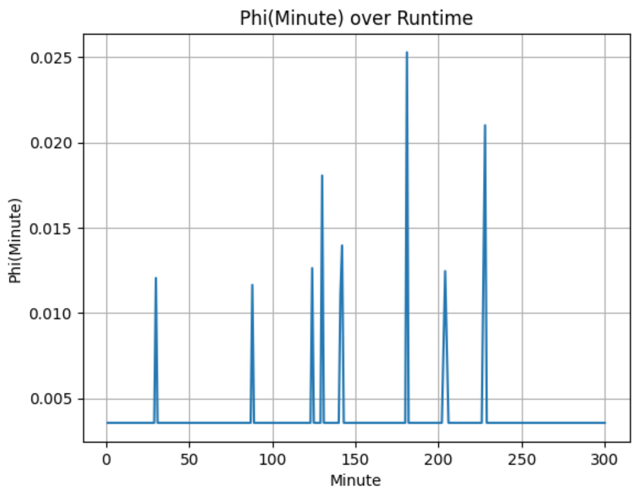
In the end, in the interest of at least producing some kind of results, 6 small BBN models were created - all six models included the "minute" feature, but otherwise included the following features:

- model1: release date, rating
- model3: theatrical and digital release types
- model4: comedy and drama genres
- model5: Walt Disney and Warner Bros. studios
- model6: release countries of USA and South Korea
- model7: English and Korean languages

Model2 was a version kept in the code but commented out because it was too large, hence the jump from model1 to model3. Each of these models was generated to answer specific questions about the included features. The goal of model1 was to predict runtime based on rating or release date. The model showed that, generally speaking, longer runtimes corresponded to higher ratings, while looking at runtime and release date for increases or decreases in runtime trends across date ranges showed stability in average runtimes across the last twenty years. This was supported by some of our early regression modeling. Model3 intended to answer questions about whether digital release format is the new straight-to-tv release type or whether movies that see a digital release instead of a theatrical release are competing with theatrical release movies as part of the high-budget, premium content marketed by streaming services. Model4 was meant to be used as a kind of control model - it was used to test the popular conception that comedies are shorter, and dramas are longer. Model5 was built to look at two of the larger, older, and more well-known studios, Walt Disney and Warner Bros. The question this model was meant to answer is whether Disney movies tended to be shorter than Warner Bros. because much of Disney's output is animated and the hypothesis was that live action movies have longer runtimes than animated ones. Models 6 and 7 aimed to look at whether cultural aspects influence runtime and how this could be used by filmmakers to craft their film length towards their target audience.

However, models 3 through 7 all produced implausible results for one out of two queries put to each of them. Only model1 produced entirely acceptable results. It should be noted that the features included in models 3 through 7 were all fragmented features in the one-hot encoding scheme - release type, genre, studio, release country, and language were all features that got broken out into one column for each unique value present in the columns of the original dataset. This confirmed that between the hardware limitations restricting the number of features that can be included in the BBN and the encoding scheme breaking features out into many new features, the BBN proved an ineffective model for most of the processed dataset. That said, future work could involve going back to the original dataset and producing a second working dataset with different preprocessing and cleaning techniques applied - in such a case, the BBN could still be a viable model for the data even given the hardware limitations.

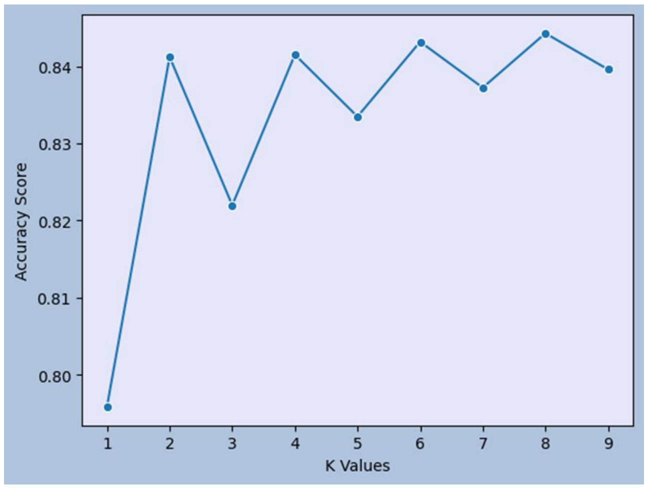
As mentioned above, model1 showed some promise despite modeling a much-reduced dataset. This model serves well to help determine the most desirable runtimes to achieve the desired rating for a film. This is shown in the plot below, which displays the most probable runtimes to achieve a 4.23 out of 5 rating.



To evaluate the usefulness of model1, F1 score was brought to bear. F1 score is a mean value of the precision and recall values, often used because it provides a balanced evaluation of the model's performance. An F1 score will be a value between 0 and 1, and it is generally accepted that F1 scores of 0 to 0.3 indicate poor model performance, 0.3 to 0.7 indicate fair performance, and 0.7 to 1 indicate excellent model performance. After performing the necessary calculations, model1 had an F1 score of 0.54, falling squarely in the fair range and indicating that results may be usable, but there is still a good deal of room for improvement in the model. This was not a surprising result, as many issues were noted in how the model could be improved even before evaluation.

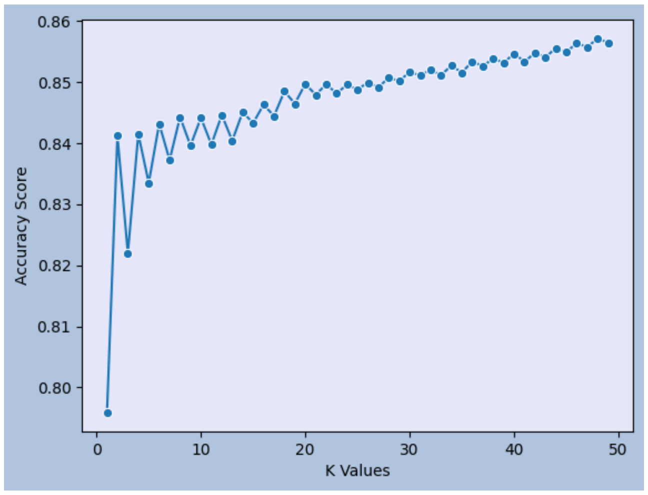
KNN Modeling

To view the data in a KNN model a binary metric needed to be created. For this the likelihood that a movie exceeds 120 mins was used as the metric. This modelling could potentially allow filmmakers to pose questions about their demographic and the historic expectations for films in specific genres, regions, or even studios. The question posed here is if one can accurately model movie length based on the given parameters.



As can be seen in the plot above, the accuracy score seemed to improve every other run, therefore theoretically the more times it is run, the more accurate results can be achieved until diminishing returns are witnessed on processing time. Outliers were removed in order to further improve prediction, but this actually had a negative effect on the results. This could have occurred due to the query posed for binary transformation of data and could prove effective with other queries. The initial model achieved a best accuracy of 85.7% after 10 runs.

To improve accuracy, the model was run again with 50 runs:



The incremental improvement every other run is at first noticeable but provided diminishing returns overtime. After 50 runs the model was able to achieve an accuracy of 0.8675, a precision score of 0.7149, and a recall of 0.5273. The accuracy could be increased by approximately 1%, however the cost of computing time was five-fold.

The KNN model also proved that there is a strong enough correlation between the independent and dependent variables to effectively predict movie runtime based on the given variables.

Summary

In conclusion some effective models were built regarding the prediction of ideal runtimes for films. Each of the models had some restrictions implemented that make them far from universally usable. However, they each could be used to get information for production companies that may lead to better received films.

For example, the BBN was one of the most effective of the model types used. This is true despite the many limitations noted above because the form of the BBN's output was directly and immediately useful in answering the questions posed in this project. Future work would focus more on this model type, as some of the current drawbacks of this iteration could be overcome by producing a working dataset with different cleaning, preprocessing, and encoding applied.

The Random Forest Model was also one of the most accurate models built. This model was built with specific restrictions on the data set used to build the model. However, even with these limitations the model would be beneficial to some of the largest production companies for determining the ideal runtimes of movies. There were also limitations on which languages and countries of origin the model was good at predicting with. Overall, this model is useful, even if it does not rely on the entire large data set.

Regarding the KNN model there was no binary variable, so logical prompts were used to create a variable. One of the primary prompts used in this study was whether the runtime was greater than 120 minutes. This approach is interesting as it allows any variable within the data to be viewed as the dependent variable and for prompting the model with more complex questions. While the model achieves a high score with simplified logical prompts, increasing the complexity of a given prompt is hypothesized to negatively impact the F1 score.

Overall, there are strong accuracy and F1 scores across the models, this indicates a strong correlation of the independent variables to our dependent ‘runtime’ variable. All the models indicated fair performance in modeling expectations, some even achieving high performance such as KNN, BBN, and Random Forest Regression Model. This provides strong evidence to support the conclusion that movie runtime can indeed be predicted based on factors such as release date, country, language, studio, etc.

Applications

BBN (Bayesian Belief Network)

Arguably the most useful potential applications of knowledge gained from the built models comes from the Bayesian Belief Network model that showed conditional probabilities for most likely runtime given a certain rating. This information can be used by filmmakers to add or cut scenes from their movies, allowing them to craft films that are more likely to be received well by fitting them within a range of desirable runtimes. For example, using this type of modelling, a filmmaker could collect data on the most probable runtimes for movies across ratings from 4 to 5 out of 5, producing a range of runtimes that are most likely to keep audiences engaged without getting bored after too long.

Another useful application of knowledge from the BBN model type is how cultural aspects influence runtimes, again allowing filmmakers to craft their films with intention based on the audience they are trying or expecting to reach. Using the BBN to predict the average length of movies that French audiences enjoy most using the runtime, rating, release country, and language features is important knowledge for a filmmaker who plans to release a French language film in France and Canada but not in the US, for instance. It could also allow filmmakers to tailor their films to audiences in different regions - in the past, film length has tended to affect release type, with shorter versions of a movie seeing theatrical release regardless of market and extended versions only seeing physical or digital release later, again regardless of market. Applying insights from our models, a filmmaker in the above-described scenario might discover that Canadians tend to rate longer movies higher, while French citizens rate shorter films more highly, and might therefore release the extended cut to Canadian audiences in theaters and the shorter cut to their French audience at the same time.

KNN (K-Nearest Neighbor)

As with the BBN model KNN modeling allows the posing of questions in binary queries, such as “is rating greater than 4 if min is greater than 200”, or “if studio x, y, or z then is min greater than 120?” By posing such questions filmmakers are offered greater insight into what a norm for their audience is, providing them the information they need to conform to expectations or break away from them completely.

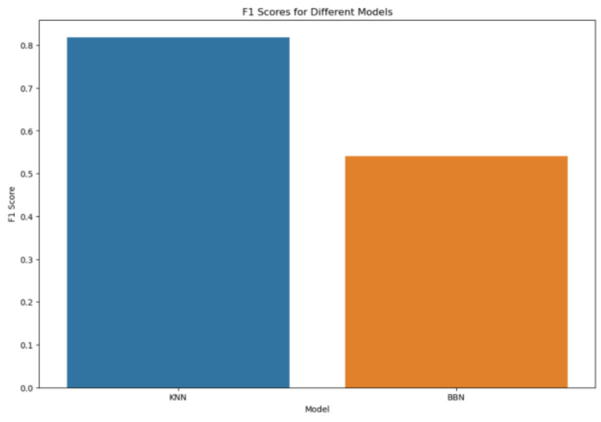
Random Forest Regression

Using random forest regression would not provide accurate predictions as some of the other models laid out in this document. However, one could use it to predict movie lengths, no longer than three hours, based on other certain features such as a limited pool of studios, and any genre. These predictions, using the specific model in the accompanying code, would provide accuracy in predicting runtime to within ~17 minutes.

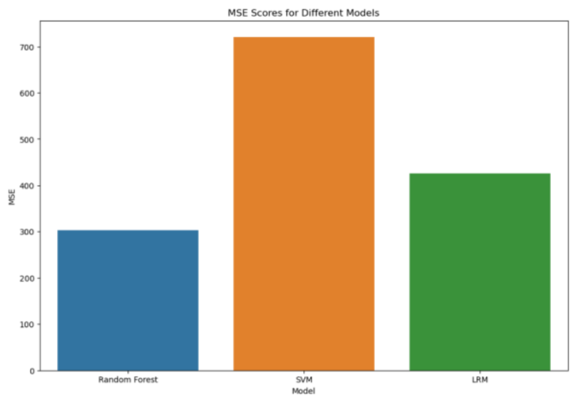
LRM (Linear Regression Model)

Continuous models such as the linear regression offer a different perspective: instead of providing binary queries, parameters such as studio, language, release date, etc. can be input and the result would show what the historical expectation would be for length. For example, to achieve at least a rating of 3 the parameters can be modeled with several ratings greater than 3 to build an understanding of how length plays a role in modeling the expectation for the various ratings based on other parameters. These models are more effective at modeling trends across time and noticing patterns, by manipulating the parameters one can effectively gain an understanding of how each decision might impact a potential film.

Visualization



The KNN model produced the best F1 score by far, to achieve this the model was initially run with k=10. Upon noticing that the score increased at a rate resembling F1 = x, it was rerun a few times to see the best score that could be achieved in a reasonable computation time, looking for under 3 minutes on a desktop. The result of the second model was an improvement of 1 percent in F1 and five-fold expense for computational time. There were further attempts to improve the model by removing outliers, but as the model was predicting if movies were likely to have a runtime greater than 120 minutes this negatively affected the F1 score. This is not to say removing outliers could not be helpful but that it is dependent on the logical prompt used for implementing a binary dependent variable. The full implementation can be viewed at: https://github.com/cstillma/MovieMining/blob/main/K-Nearest%20Neighbours.ipynb



As seen in the above graph, the Random Forest produced the smallest MSE by a large margin. This indicates that the Random Forest Regression model was more accurate than the other two models. It is important to note that this final model reduced the maximum runtime to 3 hours to achieve this goal (as mentioned previously). For further images and walkthrough of the Random Forest Regression Model specifically please see the Notebook breakdown here: https://github.com/cstillma/MovieMining/blob/main/Random_Forest.ipynb

REFERENCES

[1] James E. Cutting, 2016. *The evolution of pace in popular movies*, Cognitive Research Journal DOI: https://cognitiveresearchjournal.springeropen.com/articles/10.1186/s41235-016-0029-0
[2] acmk19. 2017. *Time Isn't Money: Does Runtime Affect a Film's Box Office?*, Film and Digital Media, DOI: https://filmanddigitalmedia.wordpress.com/2017/10/19/time-isnt-money-does-runtime-affect-a-films-box-office/
[3] *How is the success of films and TV shows measured?,* IMDBPro DOI: https://pro.imdb.com/content/article/entertainment-industry-resources/featured-articles/how-is-the-success-of-films-and-tv-shows-measured/GLFTC8ZLBBUSNTM3
[4] Letterboxd Kaggle Data. DOI: https://www.kaggle.com/datasets/gsimonx37/letterboxd/data
[5] Feature Film Wikipedia Data. DOI: https://en.wikipedia.org/wiki/Feature_film