Report

Zishuo Li, z5155437(one person project)

preface:

Since the test set and training set are larger than 2mb, please download from Google drive:
https://drive.google.com/drive/folders/1xYI3t2wDeURaWryJy3dS6tk43bMr2gOW?usp=sharing

Also, I use give update the zip file contain model.py, result file named submission.csv and Readme which also contain the same link and haw to run the project.

Introduction[from kaggle]:

Cinema is a great art and one of the most common forms of entertainment. At the same time, a good movie can bring huge economic benefits. Therefore, I hope to predict the box office of films through various characteristics of the film.

“In this competition, you're presented with metadata on over 7,000 past films from The Movie Database to try and predict their overall worldwide box office revenue. Data points provided include cast, crew, plot keywords, budget, posters, release dates, languages, production companies, and countries.use only data that would have been available before a movie's release.”[kaggle]

In this case, I need to predict an exact value, linear Regression would be a suitable algorithm, so I chose KNN for this task.

Implementation:

The source code of the project is provided with the file. Specifically, In this task, we need to find out the feature through the data in the training set, and then predict the result of the test set through the KNN model.

K nearest neighbours is a simple algorithm that stores all available cases and predicts the numerical target based on a similarity measure (e.g., distance functions). KNN has been used in statistical estimation and pattern recognition already at the beginning of the 1970s as a non-parametric technique. [CM,nearest Neighbor Pattern Classification,1967]

A simple implementation of KNN regression is to calculate the average of the numerical target of the K nearest neighbours.

Knn uses a distance function to decide what is ”nearest”.
The most commonly used distance function is the Euclidean distance, also known as the formula distance.If [x1,x2,x3....xn] and [y1,y2,y3...yn] are two points on the N-Dimensional space, then the distance between them is

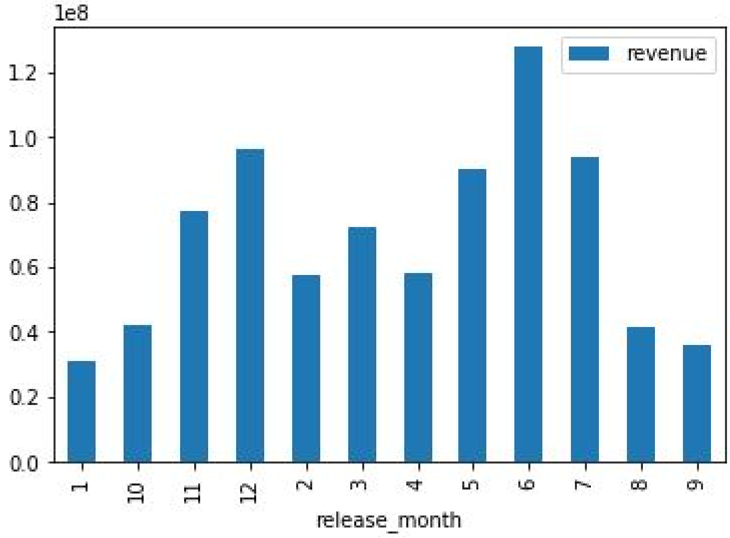
$$d_2(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

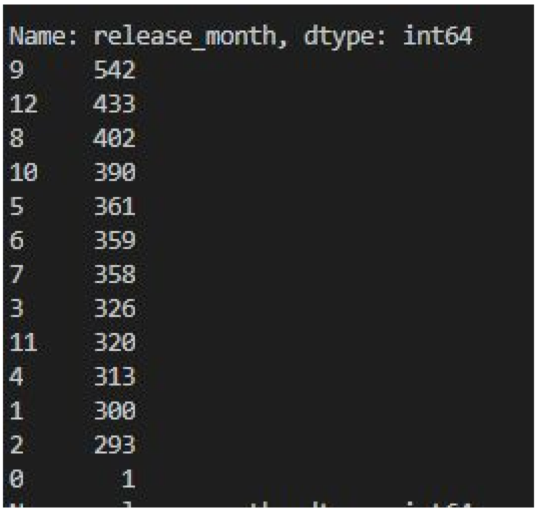
In my project I used Knn in sklearn, it uses Minkowski(usually Euclidean metric) to calculate the distance.

Feature

First, I clear data and find features in the test set. I choose `'budget','original_language','popularity','runtime','status','production_companies','production_countries','spoken_languages'`

also, The release_mont of films also affects the box office, So I add it in train_set.





```
Name: release_month, dtype: int64
9     542
12    433
8     402
10    390
5     361
6     359
7     358
3     326
11    320
4     313
1     300
2     293
0       1
```

Then convert the categorical variable into dummy/indicator variables. Dummy variables are useful because they enable us to use a single regression equation to represent multiple groups. This means that we don’t need to write out separate equation models for each subgroup. The dummy variables act like ‘switches’ that turn various parameters on and off in an equation.

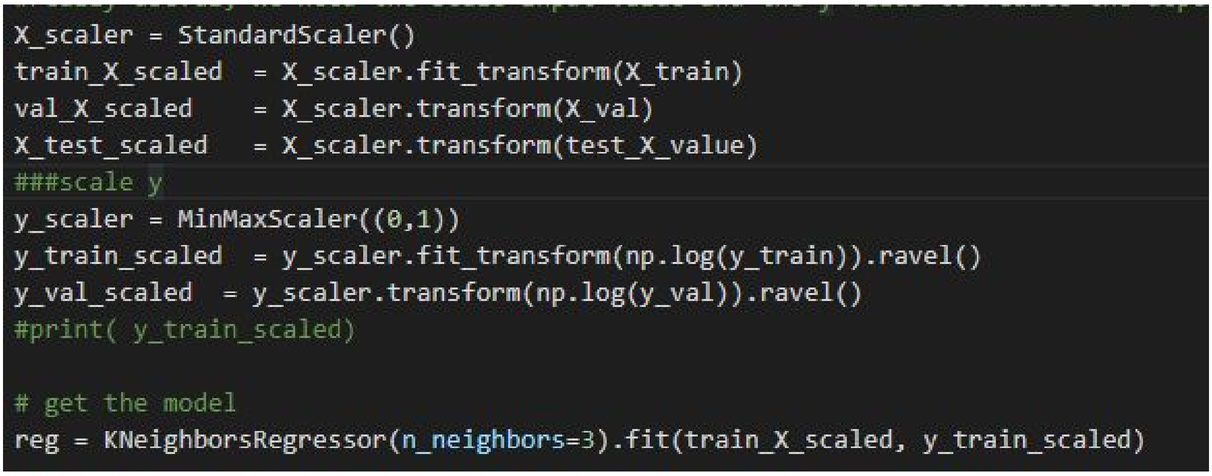
Data_nomalization
Then for the dummied data_set, I need to scale the value. Feature scaling (also known as data normalization) is the method used to standardize the range of features of data. Since the range of values of data may vary widely, it becomes a necessary step in data preprocessing while using machine learning algorithms.

In KNN, this is very important. Values that have not been normalized will lead to a significantly larger or smaller distance ratio of one axis of the function, resulting in unreasonable classification results and It makes the regression difficult to fit. Therefore, we need to normalize all values to reduce errors.

Standardize features by removing the mean and scaling to unit variance

The standard score of sample x is calculated as:
   z = (x - u) / s

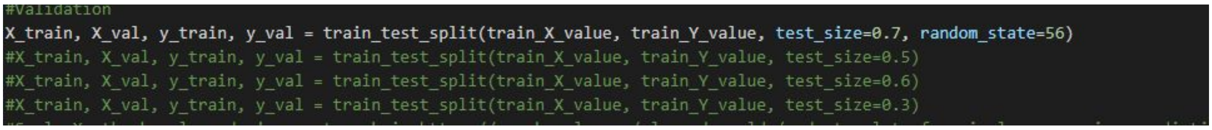
where u is the mean of the training samples or zero if with_mean=False and s is the standard deviation of the training samples or one if with_std=False.



```
X_scaler = StandardScaler()
train_X_scaled  = X_scaler.fit_transform(X_train)
val_X_scaled    = X_scaler.transform(X_val)
X_test_scaled   = X_scaler.transform(test_X_value)
###scale y
y_scaler = MinMaxScaler((0,1))
y_train_scaled  = y_scaler.fit_transform(np.log(y_train)).ravel()
y_val_scaled  = y_scaler.transform(np.log(y_val)).ravel()
#print( y_train_scaled)

# get the model
reg = KNeighborsRegressor(n_neighbors=3).fit(train_X_scaled, y_train_scaled)
```

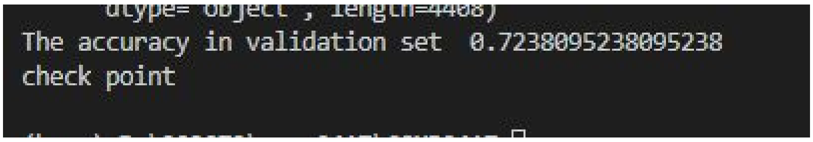
cross-validation
Through cross-validation, I found that the model performed best when the number of neighbours was equal to 3,   therefore, in this Knn model, k = 3.



```
#Validation
X_train, X_val, y_train, y_val = train_test_split(train_X_value, train_Y_value, test_size=0.7, random_state=56)
#X_train, X_val, y_train, y_val = train_test_split(train_X_value, train_Y_value, test_size=0.5)
#X_train, X_val, y_train, y_val = train_test_split(train_X_value, train_Y_value, test_size=0.6)
#X_train, X_val, y_train, y_val = train_test_split(train_X_value, train_Y_value, test_size=0.3)
```

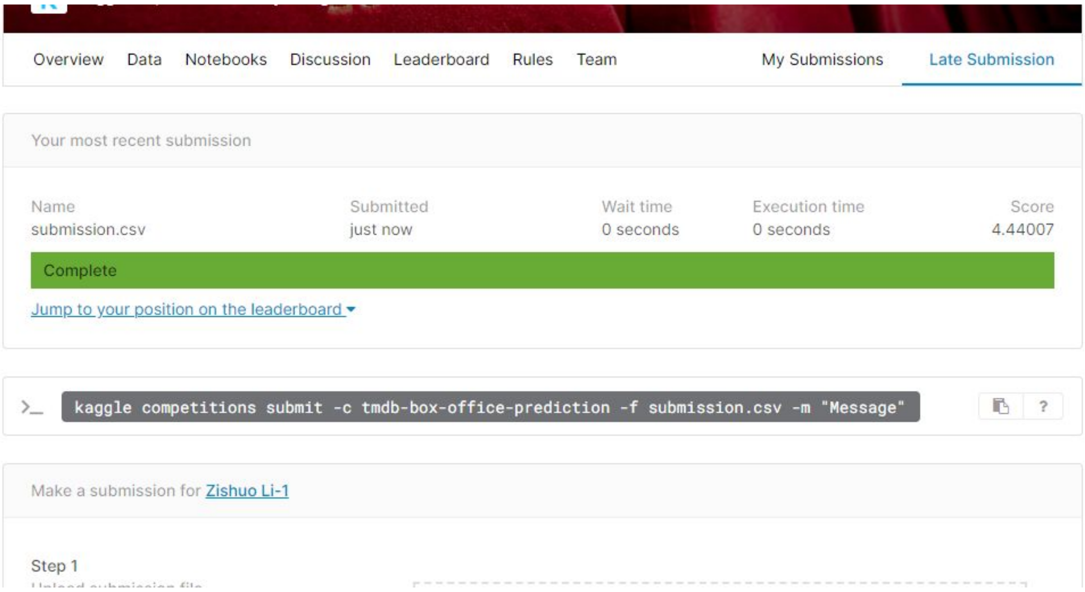
Result:

After running it, I got the accuracy on the validation_set, because the movie box office is a very large value, and prediction can't be completely the same with true box office. I assumed that when the error is less than 30%, Knn prediction of the movie box office is accurate. I get the result with an accuracy of 72% after several tests.



```
    dtype='object', length=4408)
The accuracy in validation set  0.7238095238095238
check point
```

Do prediction to test_set, and update it to kaggle.



Your most recent submission

Name	Submitted	Wait time	Execution time	Score
submission.csv	just now	0 seconds	0 seconds	4.44007

Complete

Jump to your position on the leaderboard ▾

```
kaggle competitions submit -c tmdb-box-office-prediction -f submission.csv -m "Message"
```

Make a submission for Zishuo Li-1

Step 1

Upload submission file

conclusion:

As a simple algorithm, KNN does not perform very accuracy result, and sometimes overfitting occurs. However, as a single-person project, KNN algorithm can provide an intuitive and easy way to complete tasks, and performs well on such linear tasks. Therefore, it can be said that KNN is an algorithm that can be used in this box office prediction project

Future Work:

Next, I need to further improve the feature engineering, find more features from the data set, and use a more effective algorithm, such as EDA, which is more suitable for this task.

Reference:

scikit-learn[https://scikit-learn.org/stable/index.html]
[T.M. COVER, nearest Neighbor Pattern Classification Np.1, JANUARY 1967]
[https://www.kaggle.com/c/tmdb-box-office-prediction/overview]