داده کاوی روی مجموعه داده المپیک ۲۰۲۱ توکیو

مریم واقعی*

اطلاعات گزارش

تاریخ:
۱۶ بهمن ۱۴۰۰

واژگان کلیدی:
داده کاوی
زبان برنامه نویسی پایتون
پیش پردازش داده ها
الگوریتم های بی نظارت
الگوریتم های با نظارت
الگوریتم Kmeans
الگوریتم DBScan
شبکه عصبی
الگوریتم SVM
الگوریتم KNN
درخت تصمیم
الگوریتم bayes
الگوهای پرتکرار
قوانین انجمنی

چکیده

در این گزارش قرار است پروژه ی خود در درس داده کاوی را با تکیه بر دانش آموخته شده در کلاس و ابزارها و کتابخانه های زبان برنامه نویسی پایتون، تشریح کنیم. این پروژه به دو بخش فاز اول که شامل شناخت داده و پیش پردازش روی داده ها و فاز دوم شامل ساخت مدل ها و ارزیابی آنها و همچنین پیدا کردن الگوهای پرتکرار و قوانین انجمنی تقسیم می‌شود.

ما در این پروژه برای ارزیابی مدل های بدون نظارت از روش بصری سازی خوشه ها و برای مدل های با نظارت از دو روش ارزیابی accuracy score و k-fold استفاده کردیم. همچنین در مدل های با نظارت، برای درک بهتر میزان درستی و نادرستی پیش بینی کلاس ها ماتریس confusion برای داده های آموزش و داده های تست را رسم کرده ایم.

* پست الکترونیکی: mwaghei۷۸@gmail.com

فهرست مطالب

مقدمه

در گذشته و در عصری که اینترنت و تکنولوژی برای بشریت معنا نداشت داده ها و دانش و اطلاعات صرفا در ذهن انسان ها، روی دیوار غارها، برگ درختان، پوست حیوانات و بعدها روی کاغذ ذخیره می‌شد، اما امروزه با پیشرفت تکنولوژی ما در دنیایی از داده ها غرق شده ایم و داده ها به جای هک شدن بر روی دیوار غارها و برگ درختان امروزه در گوشی های همراه، کامپیوتر های شخصی، لپ تاپ ها و سرور ها ذخیره شده اند و زندگی ما انسان ها آنچنان به این داده ها وابسته شده که گویی اگر روزی تمام این داده ها و اطلاعات از بین بروند عملا ما انسان ها به عصر انسان های اولیه بازگشته ایم و زندگی ما به اندازه همان زمان سخت و طلاقت فرسا خواهد شد.

تمام این ها نشان می‌دهد که چه میزان داده ها در زندگی انسان ها نقش حیاتی و مهمی دارند و بنابراین باید این داده ها را حفظ کرد و علاوه بر حفظ آنها، از این داده ها دانش جدید بدست آورد و در اینجاست که داده کاوی و استخراج اطلاعات اهمیت و معنا پیدا می‌کند.

داده کاوی این قابلیت را ارائه می دهد که داده ها را با دیدی جدید مشاهده کنیم، و ارتباط ها و الگوهایی را که قبلاً مورد توجه نبوده اند، کشف کنیم.

ما در این پروژه سعی کردیم تا با تکیه بر ۴ مرحله اصلی داده کاوی یعنی جمع آوری داده، پیش پردازش داده، داده کاوی و درنهایت ارزیابی الگو ها بتوانیم در نهایت به مدل هایی کارا برای مجموعه داده های خود برسیم.

در فاز جمع آوری داده ما از مجموعه داده ی المپیک توکیو ۲۰۲۱ استفاده کردیم[۲]. سپس در فاز پیش پردازش عملیات پاکسازی داده ها شامل حذف افزونگی ها، پر کردن داده های از دست رفته، کاهش ابعاد در سطح ویژگی و داده را انجام دادیم تا داده ها برای مرحله بعد یعنی فاز داده کاوی آماده شوند. ما در فاز داده کاوی با اعمال الگوریتم های مختلف داده کاوی از جمله الگوریتم های kmeans و dbscan برای داده های بدون برچسب و الگوریتم های mlp و svm و knn و bayes و decision tree و ensemble برای داده های دارای برچسب سعی کردیم تا مدل هایی را متناسب با داده های خود ایجاد کنیم و درنهایت نیز در فاز ارزیابی ما با روش های ارزیابی با محاسبه ی دقت با متریک های accuracy و k-fold، دقت مدل ها را مورد بررسی و ارزیابی قرار دادیم.

[۲] مجموعه داده ی المپیک ۲۰۲۱ در سایت kaggle در دسترس می باشد.

بخش اول: آشنایی با مجموعه داده

ما در این پروژه از مجموعه داده ۲۰۲۱_olympics استفاده کرده ایم که منبع اصلی این دیتاست سایت kaggle [۳] می باشد.
این دیتاست شامل ۵ فایل با نام های زیر است:

۱. Athletes.xlsx
۲. Coaches.xlsx
۳. EntriesGender.xlsx
۴. Medals.xlsx
۵. Teams.xlsx

در فایل Athletes.xlsx اطلاعات ورزشکاران گزارش شده که این شامل سه ویژگی نام ورزشکاران، نام کشورشان و رشته ورزشی آنها می باشد. این فایل ۱۱۰۶۲ ردیف منحصر به فرد دارد.

About this table

Details about Athletes

Name	NOC	Discipline
11062 unique values	United States of Am... 6% Japan 5% Other (9884) 89%	Athletics 19% Swimming 7% Other (8274) 75%
AALERUD Katrine	Norway	Cycling Road
ABAD Nestor	Spain	Artistic Gymnastics
ABAGNALE Giovanni	Italy	Rowing
ABALDE Alberto	Spain	Basketball
ABALDE Tamara	Spain	Basketball

تصویر ۱- جزئیات اطلاعات ستون ها در فایل Athletes.xlsx

در فایل Coaches.xlsx اطلاعات مربیان تیم ها از جمله نام آنها، نام کشور و نام رشته ورزشی و همچنین رویدادی که در آن حضور دارند را نشان میدهد. در این فایل یکسری مقادیر null در ستون events وجود دارد که در فاز اول مقادیر آن را پر خواهیم کرد. این فایل همچنین شامل ۳۸۱ ردیف منحصر به فرد می‌باشد.

About this table

Details about coaches, countries and disciplines along with event

Name	NOC	Discipline	Event
381 unique values	Japan 9% Spain 7% Other (331) 84%	Basketball 19% Artistic Swimming 18% Other (251) 64%	[null] Men Other (155)
ABDELMAGID Wael	Egypt	Football	
ABE Junya	Japan	Volleyball	
ABE Katsuhiko	Japan	Basketball	
ADAMA Cherif	Côte d'Ivoire	Football	
AGEBA Yuya	Japan	Volleyball	

تصویر ۲ - جزئیات اطلاعات ستون ها در فایل Coaches.xlsx

[۳] https://www.kaggle.com/arjunprasadsarkhel/۲۰۲۱-olympics-in-tokyo

فایل EntriesGender.xlsx اطلاعاتی در مورد تعداد خانم ها و آقایان و تعداد کل ورزشکاران در هر رشته ورزشی را نشان می دهد که این فایل شامل ۴۲ ردیف منحصر به فرد میباشد.

About this table

Details about the Discipline and the number of females and males participating

Discipline	Female		Male		Total
46 unique values	98 86 Other (42)	4% 4% 91%	144 0 Other (41)	7% 4% 89%	288 96 Other (42)
3x3 Basketball	32		32		64
Archery	64		64		128
Artistic Gymnastics	98		98		196
Artistic Swimming	105		0		105
Athletics	969		1072		2041

تصویر ۳ – اطلاعات ستون ها در فایل EntriesGender.xlsx

فایل Medals.xlsx اطلاعاتی در مورد نام کشور و تعداد مدال های طلا، نقره و برنز و همچنین رتبه هر کشور به ما میدهد. این فایل شامل ۹۳ ردیف منحصر به فرد می‌باشد.

About this table

Contains the Medals and Scoreboard of countries that participated in Tokyo Olympics as of 26th July 2021

Rank		Team/NOC	Gold		Silver
86 77 Other (79)	9% 6% 85%	93 unique values	0 1 Other (43)	30% 24% 46%	1 0 Other (43)
1		United States of America	39		41
2		People's Republic of China	38		32
3		Japan	27		14
4		Great Britain	22		21
5		ROC	20		28

تصویر ۴ – اطلاعات ستون ها در فایل Medals.xlsx

در نهایت در فایل Teams.xlsx ما اطلاعاتی شامل نام تیم، رشته ورزشی، نام کشور و رویدادی که آن تیم در آن شرکت میکند را مشاهده میکنیم. این فایل نیز شامل ۷۴۳ ردیف می باشد.

About this table

Details about the Teams, discipline, Name of Country and the event

Name		Discipline		NOC		Event
Japan	6%	Swimming	15%	Japan	6%	Men
United States	6%	Athletics	11%	United States of Am...	6%	Women
Other (655)	88%	Other (551)	74%	Other (648)	87%	Other (509)
Belgium		3x3 Basketball		Belgium		Men
China		3x3 Basketball		People's Republic of China		Men
China		3x3 Basketball		People's Republic of China		Women
France		3x3 Basketball		France		Women
Italy		3x3 Basketball		Italy		Women

تصویر ۵ – اطلاعات ستون ها در فایل Teams.xlsx

بخش دوم: تحلیل مجموعه داده

در ابتدا و قبل از شروع پیش پردازش چون داده های ما بیشتر از جنس رشته هستند، فلذا ما برای این ستون ها، یک ستون جدید به هر دیتافریم مربوط به هر فایل اضافه کردیم که در آن مقادیر رشته ای هر ستون را به مقادیر عددی نگاشت کردیم.

این کار به این دلیل است ما که در ادامه و در برخی مدل ها و الگوریتم ها نمیتوان از مقادیر رشته ای استفاده کرد. در نتیجه لازم است تا نگاشت مقادیر رشته ای به اعداد را داشته باشیم.

در ابتدا ما به بررسی مجموعه داده پرداختیم. ابتدا تعداد مقادیر منحصر به فرد در هر ستون را بدست آوردیم که به شرح زیر میباشد:

```
column_name number_of_unique_values
Name 381
NOC 61
Discipline 9
Event 7
Name_numeric 381
NOC_numeric 61
Discipline_numeric 9
Event_numeric 7
```

تصویر ۶- تعداد مقادیر منحصر به فرد در ستون های فایل Coaches.xlsx

```
column_name number_of_unique_values
Name 11062
NOC 206
Discipline 46
Name_numeric 11062
NOC_numeric 206
Discipline_numeric 46
```

تصویر ۷ – تعداد مقادیر منحصر به فرد در ستون های فایل Athletes.xlsx

```
column_name number_of_unique_values
Discipline 46
Female 38
Male 41
Total 41
Discipline_numeric 46
```

تصویر ۸ – تعداد مقادیر نحصر به فرد در ستون های فایل EntriesGender.xlsx

```
column_name number_of_unique_values
Rank 67
Team/NOC 93
Gold 14
Silver 17
Bronze 21
Total 30
Rank by Total 30
Team/NOC_numeric 93
```

تصویر ۹ – تعداد مقادیر منحصر به فرد در ستون های فایل Medals.xlsx

```
column_name number_of_unique_values
Name 146
Discipline 20
NOC 84
Event 36
Name_numeric 146
Discipline_numeric 20
NOC_numeric 84
Event_numeric 36
```

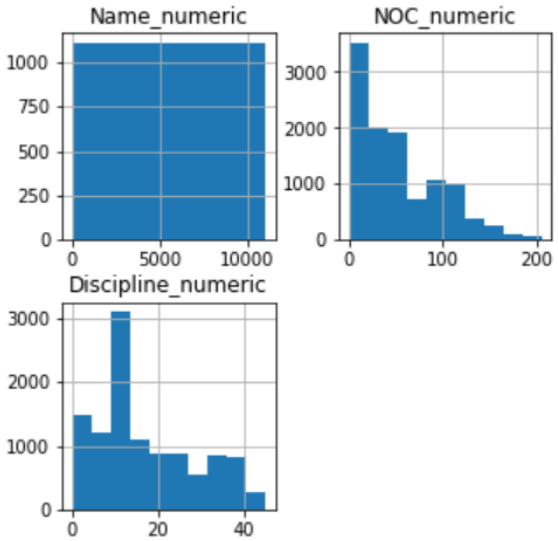
تصویر ۱۰ – تعداد مقادیر منحصر به فرد در ستون های فایل Teams.xlsx

۲,۱- خصوصیات هر ویژگی

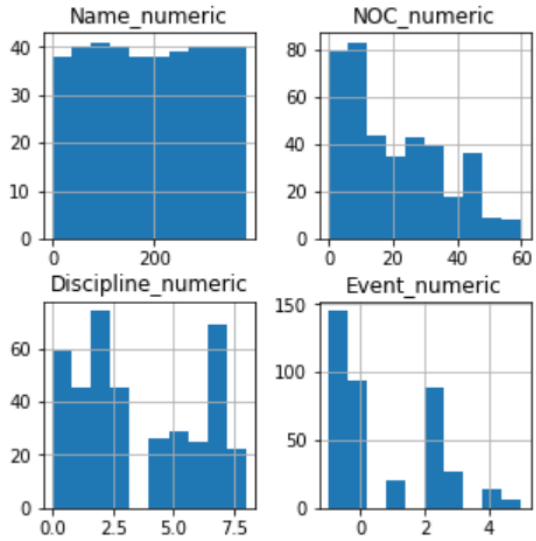
ما اطلاعات ستون ها در تمام دیتافریم ها شامل نام، نوع داده، بازه مقادیر، میانگین، میانه، مد و کمترین و بیشترین مقدار را بدست آوردیم و در متغیر دیکشنری تمام این اطلاعات را ذخیره کردیم.

۲,۲- کشیدگی داده های هر ستون به کمک نمودار هیستوگرام

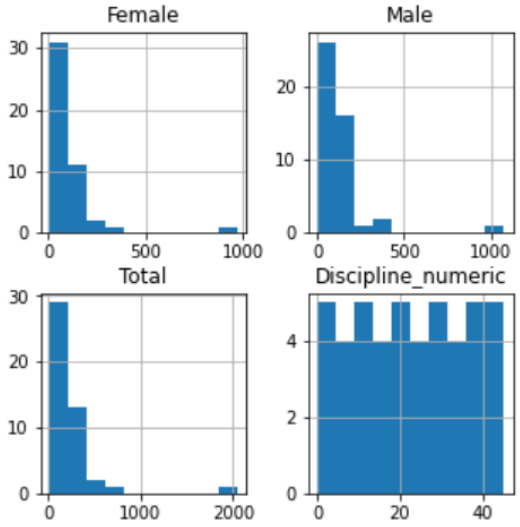
پس از آن ما نحوه توزیع داده ها و چولگی داده ها را به کمک نمودار هیستوگرام نمایش دادیم. که به صورت زیر میباشد:



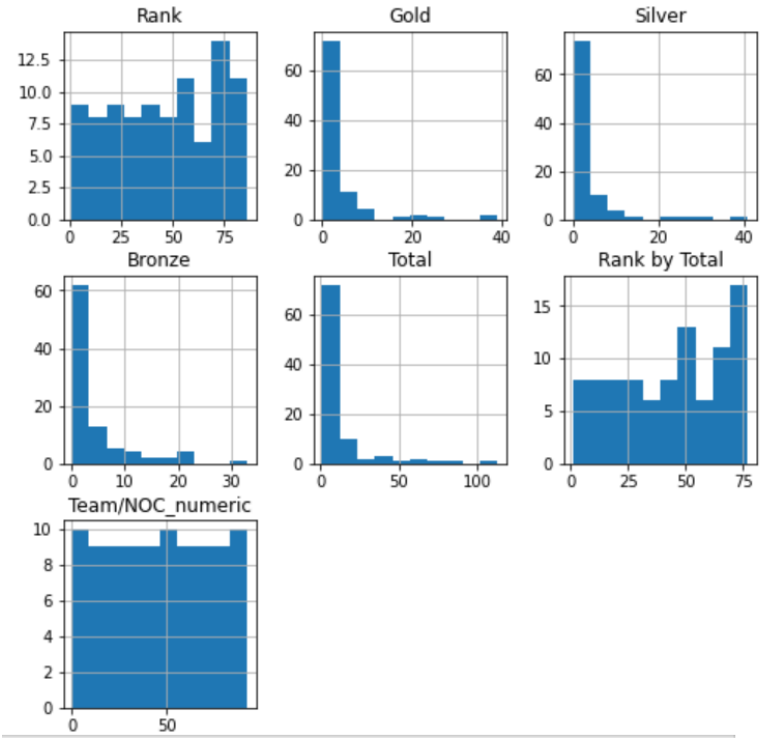
تصویر ۱۱ – نمودار هیستوگرام برای ستون های عددی دیتافریم athletes_df



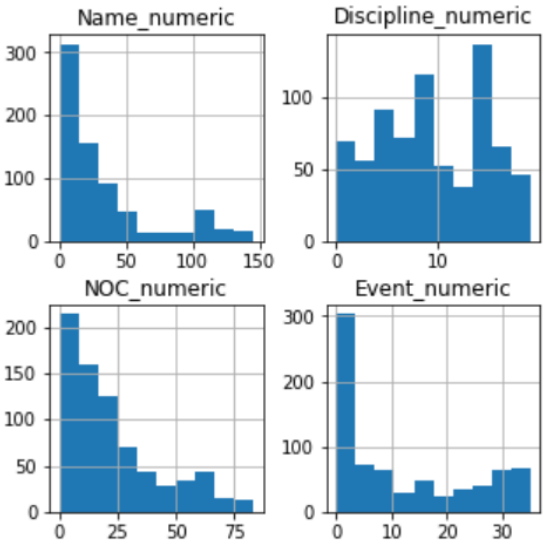
تصویر ۱۲ – نمودار هیستوگرام ستون های عددی در دیتافریم coaches_df



تصویر ۱۳ – نمودار هیستوگرام مربوط به ستون های عددی دیتافریم entries_gender_df



تصویر ۱۴ – نمودار هیستوگرام ستون های عددی در دیتافریم medals_df



تصویر ۱۵ – نمودار هیستوگرام ستون های عددی در دیتافریم teams_df

۲,۳- صحت، معتبر بودن و کامل بودن داده ها

برای چک کردن کامل بودن داده ها ما به کمک متد info در کتابخانه pandas برای هر دیتافریم بررسی کردیم که هر ستون چه تعداد داده از دست رفته داریم. که خب با بررسی که انجام دادیم متوجه شدیم تنها ستونی که مقدار از دست رفته دارد ستون Events از دیتافریم coaches_df می باشد. که مقادیر از دست رفته را در بخش های بعدی با مقادیر مناسب پر میکنیم.

برای چک کردن صحت داده ها از آنجایی که تمام ستون های عددی ما تعداد را نشان میدهند مثل تعداد خانم ها و آقایان در رشته های ورزشی یا تعداد مدال ها و... نبابراین چون نشان دهنده تعداد است و تعداد نمیتواند عدد منفی باشد پس یکی از چک هایی که روی مقادیر عددی انجام دادیم این است که این مقادیر آیا نامنفی هستند یا خیر.

مورد بعد که برای صحت مقادیر چک کردیم این است که در دیتافریم entries_gender ما تعداد خانم ها و آقایان و تعداد کل آنها را در هر رشته ورزشی داریم. بنابراین تعداد کل ورزشکاران در یک رشته ورزشی باید برابر با جمع تعداد خانم ها و آقایان در آن رشته ورزشی باشد پس این مورد را هم چک کردیم که آیا تعداد کل برابر با مجموع تعداد خانم ها و آقایان است یا خیر.

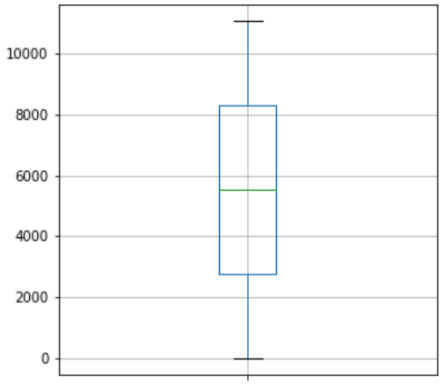
بعد از بررسی هایی که انجام دادیم برای صحت و معتبر بودن داده ها خوشبختانه ما داده نامعتبر و نادرست در دیتافریم ها نداشتیم.

۲,۴- نمودار هیستوگرام برای تمام ویژگی ها

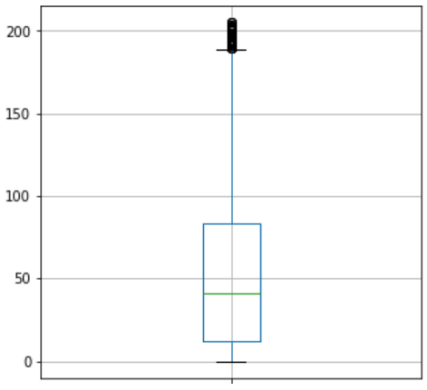
ما در بخش کشیدگی داده های هر ستون از نمودار هیستوگرام استفاده کردیم، فلذا برای مشاهده نمودار هیستوگرام هر ویژگی میتوانید به بخش ۱,۲ مراجعه کنید.

۲,۵- شناسایی داده های پرت در هر ویژگی

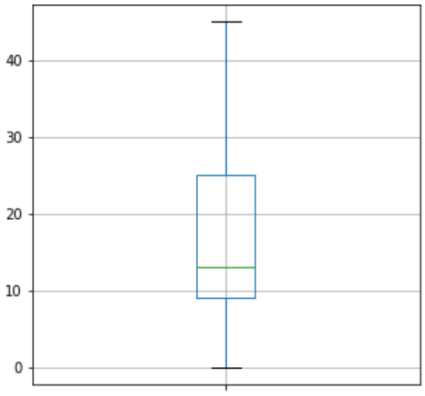
برای شناسایی داده های پرت در هر ویژگی از نمودار boxplot استفاده کردیم.



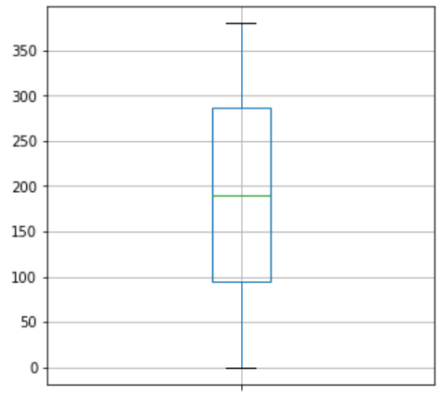
نمودار ۱- نمودارجعبه ای ستون Name_numeric از دیتافریم athletes_df



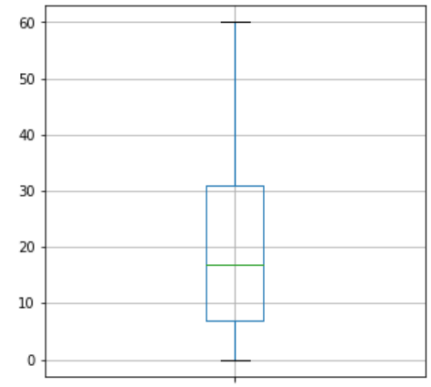
نمودار ۲- نمودار جعبه ای ستون NOC_numeric از دیتافریم athletes_df



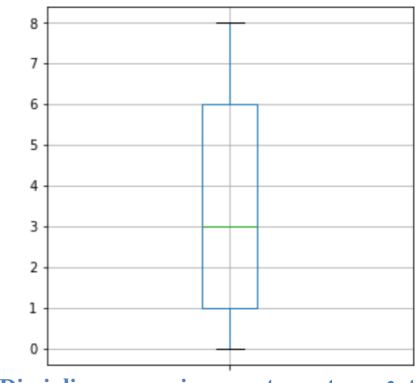
نمودار ۳ - نمودار جعبه ای ستون Discipline_numeric از دیتافریم athletes_df



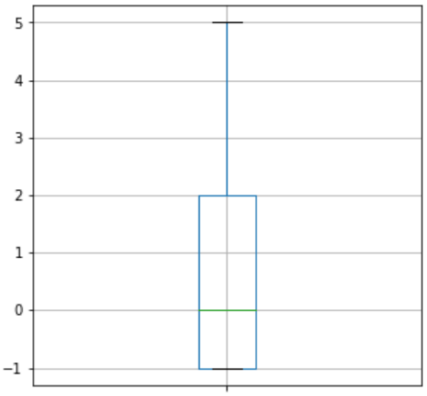
نمودار ۴ - نمودار جعبه ای ستون Name_numeric از دیتافریم coaches_df



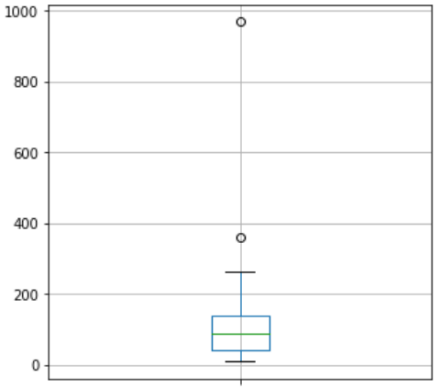
نمودار ۵ - نمودار جعبه ای ستون NOC_numeric از دیتافریم coaches_df



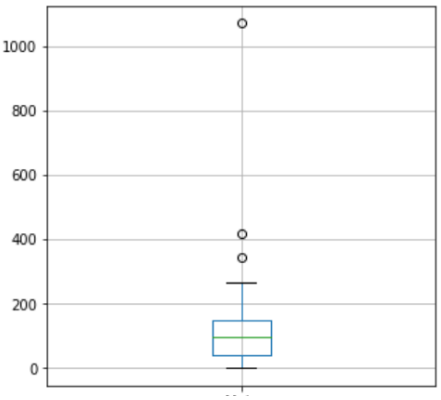
نمودار ۶ - نمودار جعبه ای ستون Discipline_numeric



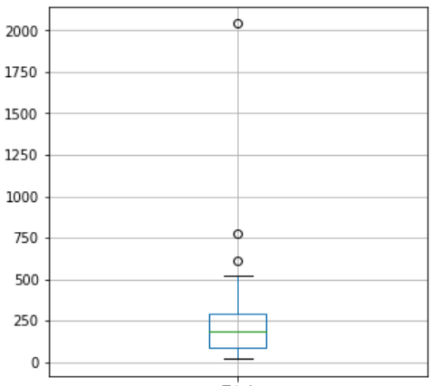
نمودار ۷ – نمودار جعبه ای ستون Event_numeric از دیتافریم coaches_df



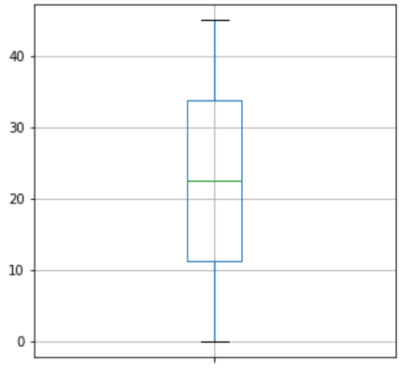
نمودار ۸ – نمودار جعبه ای ستون Female از دیتافریم entries_gender_df



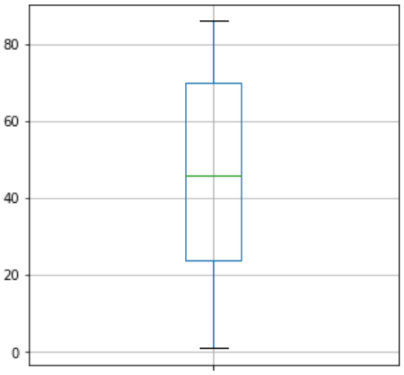
نمودار ۹ – نمودار جعبه ای ستون Male از دیتافریم entries_gender_df



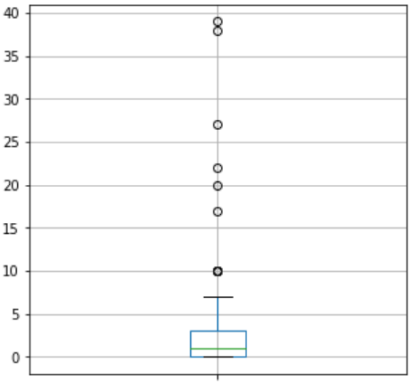
نمودار ۱۰ – نمودار جعبه ای ستون Total از دیتافریم entries_gender_df



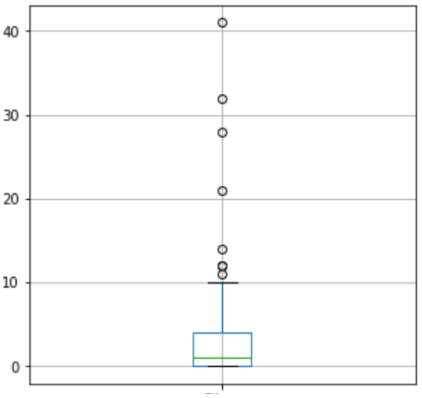
نمودار ۱۱ – نمودار جعبه ای ستون Discipline_numeric از دیتافریم entries_gender_df



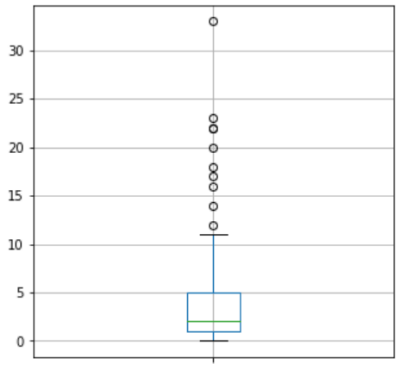
نمودار ۱۲ – نمودار جعبه ای ستون Rank از دیتافریم medals_df



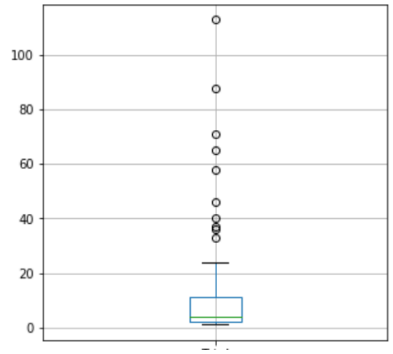
نمودار ۱۳ – نمودار جعبه ای ستون Gold از دیتافریم medals_df



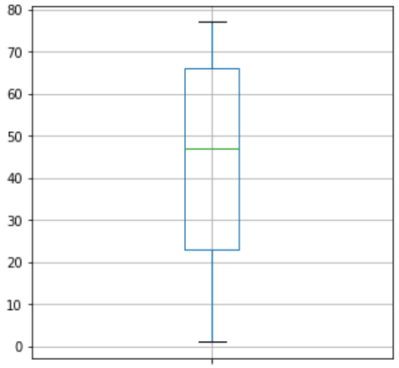
نمودار ۱۴ – نمودار جعبه ای ستون Silver از دیتافریم medals_df



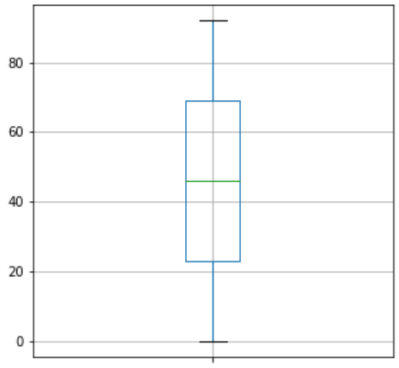
نمودار ۱۵ – نمودار جعبه ای ستون Bronze از دیتافریم medals_df



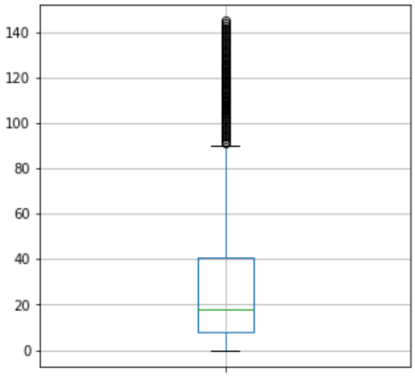
نمودار ۱۶ – نمودار جعبه ای ستون Total از دیتافریم medals_df



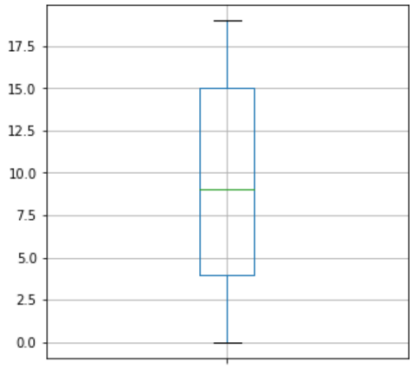
نمودار ۱۷ – نمودار جعبه ای ستون Rank by Total از دیتافریم medals_df



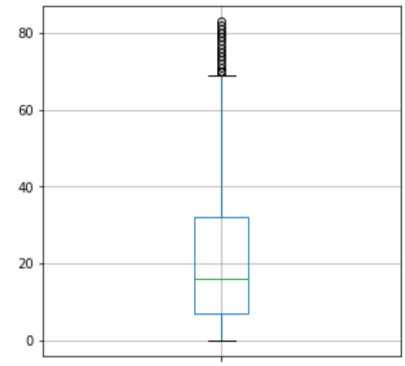
نمودار ۱۸ – نمودار جعبه ای ستون Team/NOC_numeric از دیتافریم medals_df



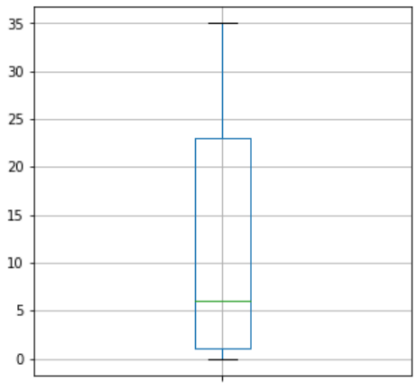
نمودار ۱۹ - نمودار جعبه ای ستون Name_numeric از دیتافریم teams_df



نمودار ۲۰ - نمودار جعبه ای ستون Discipline_numeric از دیتافریم teams_df



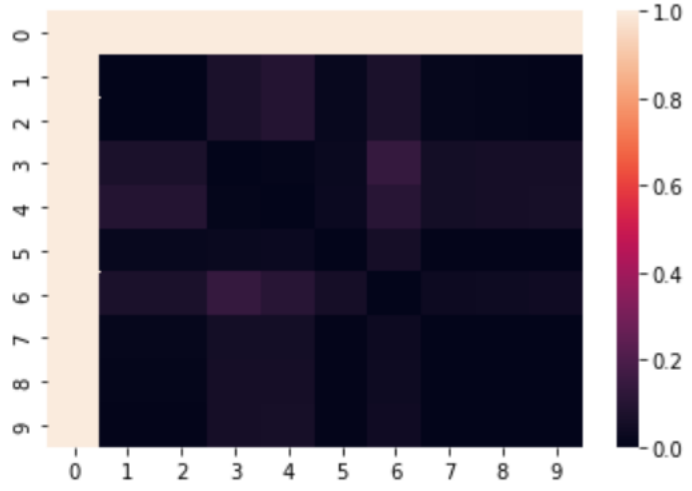
نمودار ۲۱ - نمودار جعبه ای ستون NOC_numeric از دیتافریم teams_df



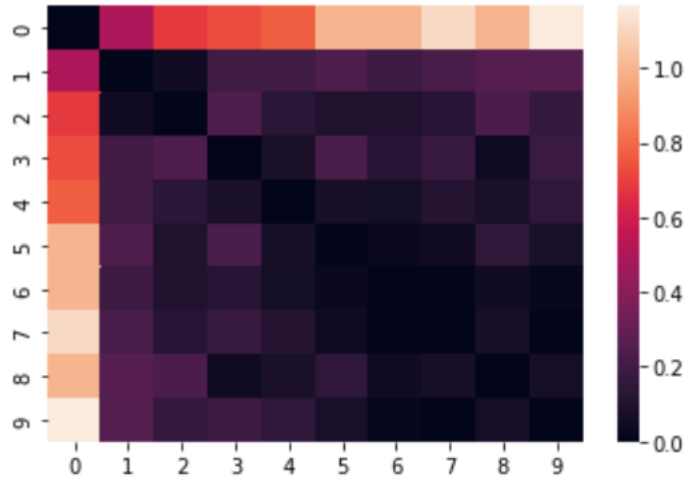
نمودار ۲۲ - نمودار جعبه ای ستون Event_numeric از دیتافریم teams_df

۲,۶- ماتریس عدم شباهت

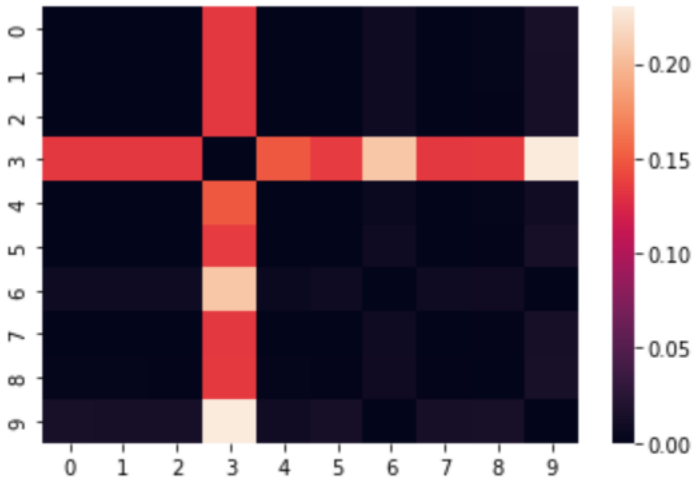
ما ماتریس عدم شباهت را با استفاده از شباهت کسینوسی برای حداکثر ۱۰ ردیف از هر دیتافریم محاسبه کردیم و مقادیر ماتریس عدم شباهت را با نمودار heatmap نمایش داده ایم که به شرح زیر است:



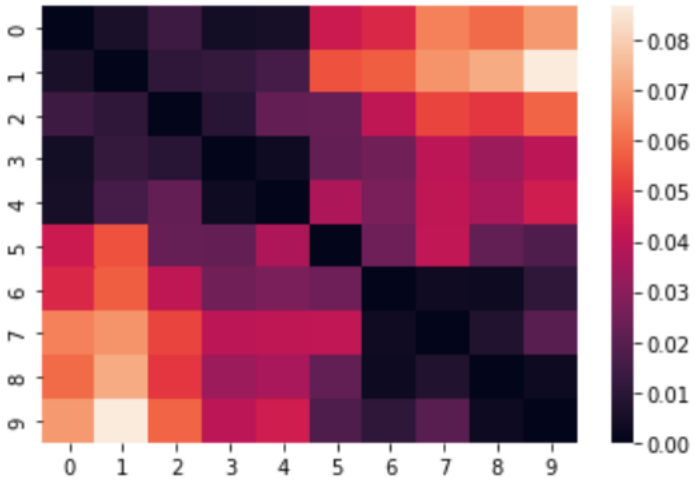
نمودار ۲۳ - ماتریس عدم شباهت ۱۰ ردیف از دیتافریم athletes_df



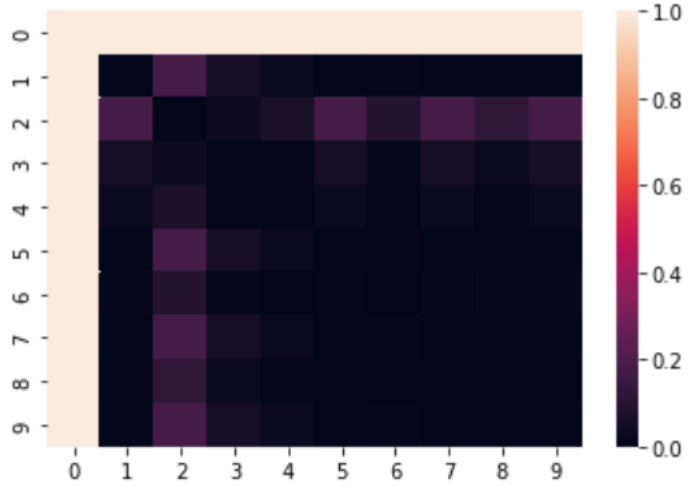
نمودار ۲۴ - ماتریس عدم شباهت ۱۰ ردیف از دیتافریم coaches_df



نمودار ۲۵ - ماتریس عدم شباهت ۱۰ ردیف از دیتافریم entries_gender_df



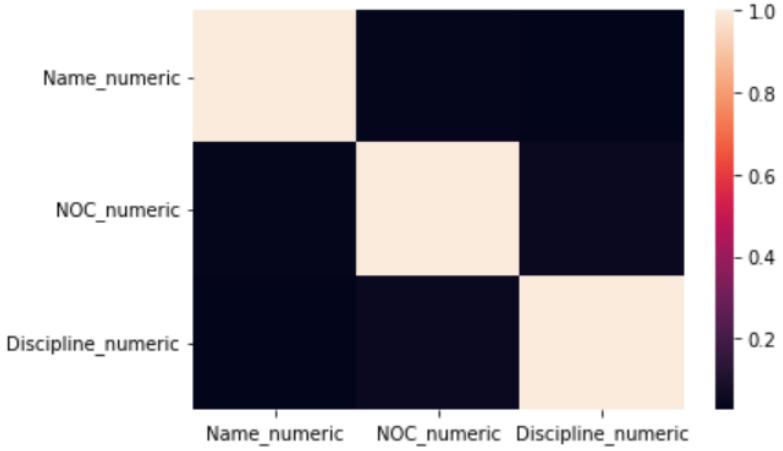
نمودار ۲۶ – ماتریس عدم شباهت ۱۰ ردیف از دیتافریم medals_df



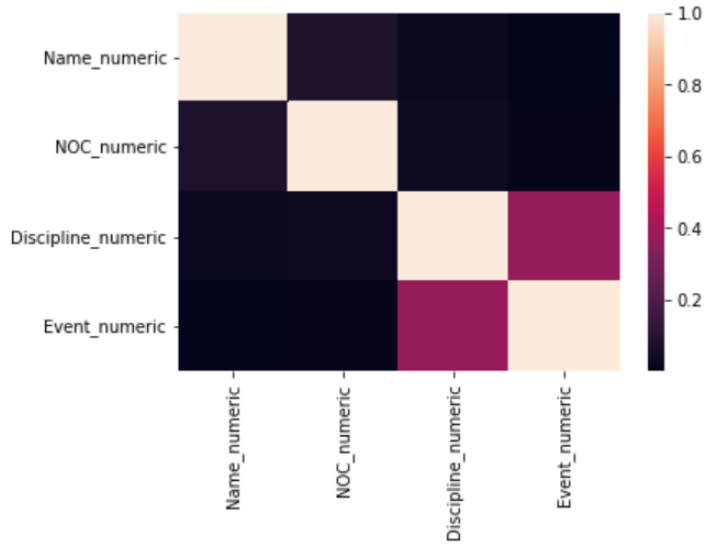
نمودار ۲۷– ماتریس عدم شباهت ۱۰ ردیف از دیتافریم teams_df

۲,۷–بررسی همبستگی ویژگی ها

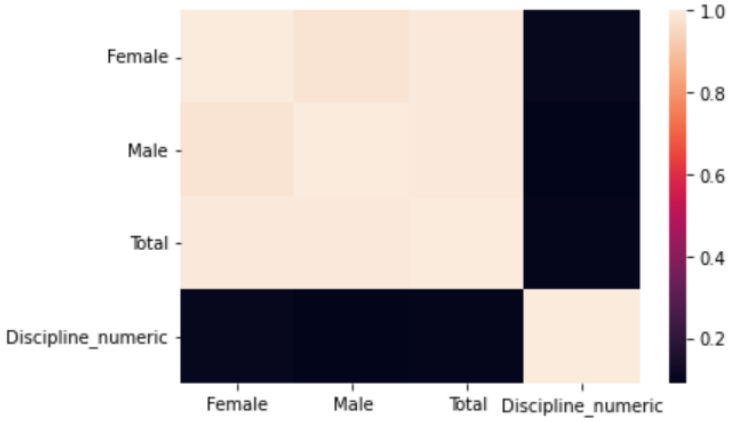
ما همبستگی ویژگی ها را به کمک معیار correlation و یا استفاده از متد corr در کتابخانه pandas بدست آوردیم و سپس با کمک نمودار heatmap همبستگی آنها را نشان داده ایم:



نمودار ۲۸ – نمودار heatmap همبستگی ویژگی ها در دیتافریم athletes_df



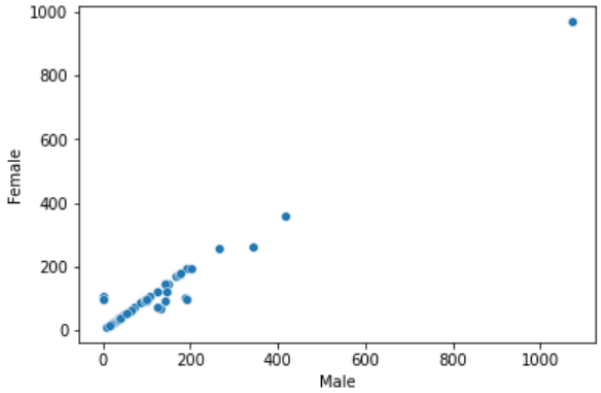
نمودار ۲۹ - نمودار heatmap همبستگی ویژگی ها در دیتافریم coaches _df



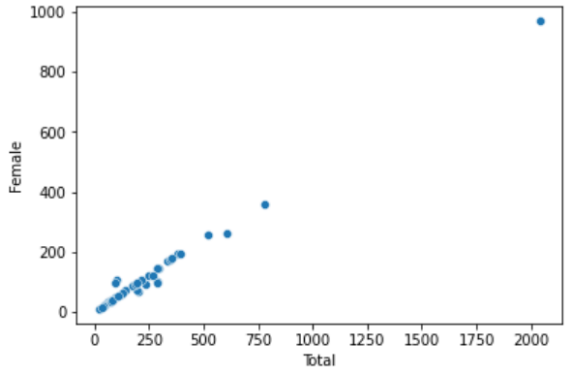
نمودار ۳۰ - نمودار heatmap همبستگی ویژگی ها در دیتافریم entries_gender_df

۲,۸-نمایش نمودار scatter برای ستون های همبسته

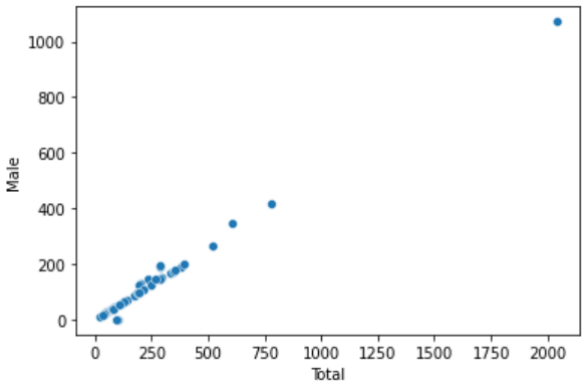
ما برای ستون هایی که مقدار همبستگی آنها بیشتر از ۰,۹۵ بوده است نمودار scatter را رسم کردیم که به صورت زیر است:



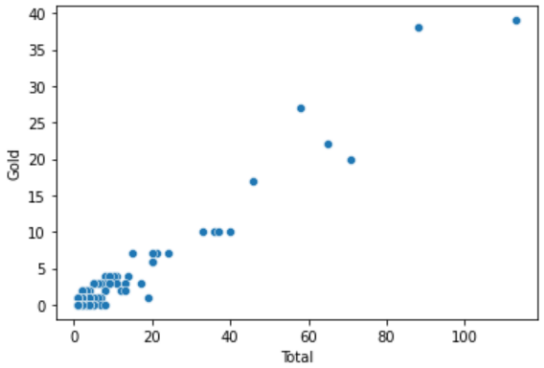
نمودار ۳۱ - نمودار scatter دو ستون Male و Female از دیتافریم entries_gender_df



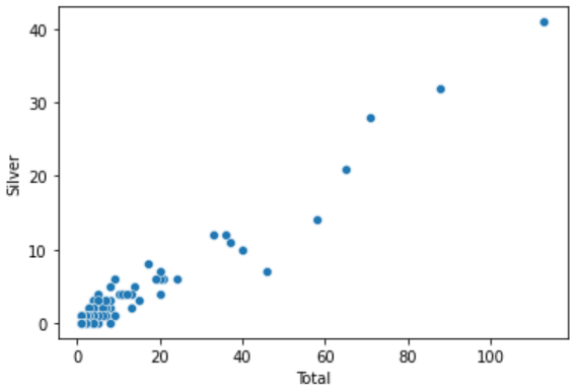
نمودار ۳۲ - نمودار scatter برای دو ستون Total و Female از دیتافریم entries_gender_df



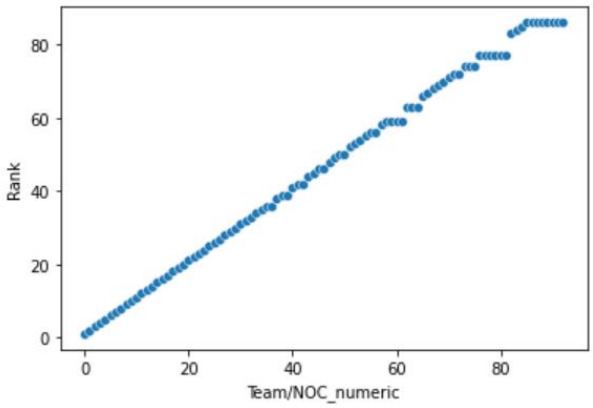
نمودار ۳۳ - نمودار scatter برای دو ستون Total و Male از دیتافریم entries_gender_df



نمودار ۳۴ - نمودار scatter برای دو ستون Total و Gold از دیتافریم medals_df



نمودار ۳۵ - نمودارscatter برای دو ستون Total و Silver از دیتافریم medals_df



نمودار ۳۶ - نمودار scatter برای دو ستون Team/NOC_numeric و Rank از دیتافریم medals_df

بخش سوم: پیش پردازش داده ها

۳,۱-پاکسازی داده ها

در بخش قبل دیدیم که تنها ستونی که داده از دست رفته دارد، ستون Event از دیتافریم coaches_df می باشد. حال ما برای اینکه مقادیر از دست رفته را پر کنیم از الگوریتم knn استفاده کردیم و داده ها را براساس ستون Events به دو دسته train و test تقسیم کردیم که در دسته train داده هایی هستند که ستون Event آن ها مقدار دارد و در دسته test داده هایی هستند که ستون Event آنها مقدار ندارد و میخواهیم مقدار آن را پیش بینی کنیم. ما نزدیکترین داده به داده ی مدنظر را در نظر گرفتیم و مقدار داده ی از دست رفته را برابر با مقدار آن ستون در داده ی نزدیک به آن قرار دادیم.

در بخش بعد برای حذف داده های پرت از دیتافریم ها، داده هایی که از q۱ - ۱,۵* IQR کمتر و از q۳ + ۱,۵ * IQR بیشتر بودند را به عنوان داده پرت در نظر گرفتیم و آن ها را حذف کردیم.

۳,۲- افزونگی در سطح ویژگی و رکورد

ما برای حذف ستون هایی که افزونگی دارند دو چیز را در نظر گرفتیم:

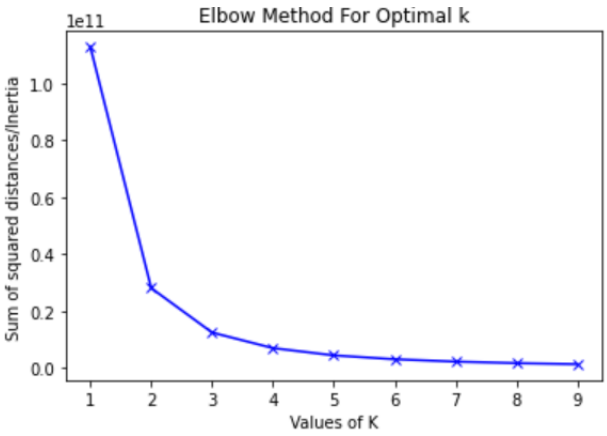
۱. برای ستون های رشته ای ما در ابتدای کار ستون های عددی را ایجاد کردیم که نگاشتی از ستون های رشته ای به اعداد هستند، پس ما اکنون نیازی به ستون های رشته ای نداریم و نگاشت عددی آنها را داریم فلذا تمام ستون های رشته ای را حذف کردیم.

۲. از بین دو ستون همبسته که مقدار همبستگی بین آنها بیشتر از ۰,۹۵ بود، نیز یک ستون را حذف و دیگری را حفظ کردیم.

برای حذف افزونگی در سطح رکورد نیز ابتدا بررسی کردیم کدام دیتافریم ها داده های تکراری دارند، سپس با استفاده از متد drop_duplicates از کتابخانه pandas، از بین ردیف های تکراری، اولین ردیف تکراری را حفظ کردیم و باقی ردیف های تکراری را حذف کردیم.

۳,۳- کاهش در سطح رکورد

برای کاهش در سطح رکورد با بررسی دیتافریم ها متوجه شدیم که همه دیتافریم ها به جز دیتافریم athletes_df تعداد داده خیلی کمی دارند. در دیتافریم athletes_df تعداد داده ۱۱۰۲۹ تا می باشد که برای کاهش تعداد از روش نمونه برداری از خوشه ها استفاده کردیم. به این ترتیب که ابتدا بهترین تعداد خوشه ها برای خوشه بندی داده های athletes_df به کمک روش kmeans را پیدا کردیم(که تعداد کلاستر مناسب آن با توجه به نمودار زیر ۳ می باشد).



نمودار ۳۷- نمودار روش Elbow که در آن مجموع مربعات فاصله ها با توجه به تعداد خوشه ها نشان داده شده.

پس از مشخص شدن تعداد مناسب خوشه ها، با روش kmeans داده های athletes_df را خوشه بندی کردیم و پس از آن به تعداد ۲۰۰۰ نمونه از هر خوشه به صورت رندوم برداشتیم.

۳,۴- نرمال سازی داده ها

برای بخش نرمال سازی داده ها ما سه روش MinMaxScaler و Normalizer از کتابخانه sklearn و zscore از کتابخانه stats استفاده کردیم. اما به این دلیل که zscore نحوه توزیع شدگی داده ها را تغییر نمیدهد و چولگی داده ها را حفظ میکند و فقط بازه ی داده ها را استاندارد سازی میکند، بنابراین تصمیم گرفتیم که از این نرمال سازی استفاده کنیم.

۳,۵- کاهش ابعاد داده ها

ما برای کاهش ابعاد داده ها از آنجایی که دیتافریم ها ۳ یا ۴ یا ۶ تا بعد داشتند لذا از روش PCA استفاده کردیم و داده ها را به فضایی بردیم که همه دیتافریم ها ۲ بعد دارند و درنتیجه ابعاد داده ها را کاهش دادیم.

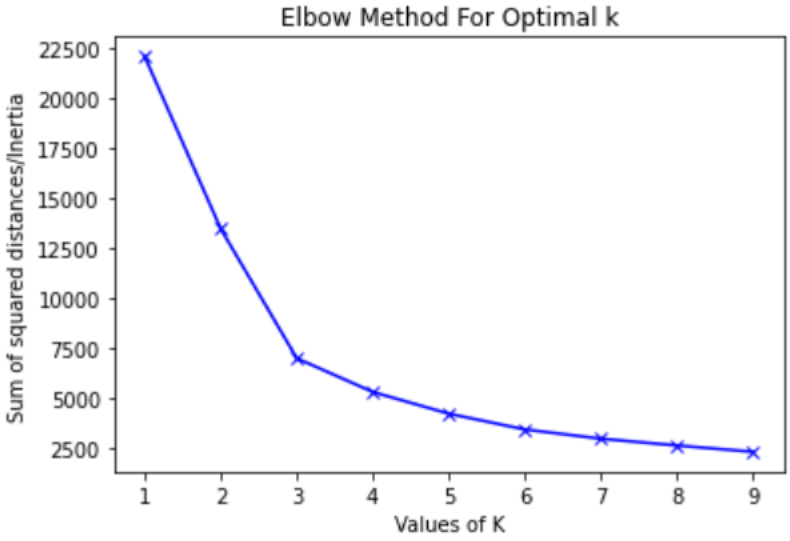
بخش چهارم: مدل سازی و داده کاوی

۴,۱- پیاده سازی الگوریتم های بی نظارت

برای این بخش ما از داده های دیتافریم athletes_df استفاده کرده ایم. به این صورت که NOC_numeric (نگاشت عددی ستون NOC) و Discipline_numeric (نگاشت عددی ستون Discipline) را برای هر ورزشکار داریم و میخواهیم بر اساس مقدار این دو ویژگی برای ورزشکاران آنها را خوشه بندی کنیم.

۴,۱,۱- الگوریتم kmeans

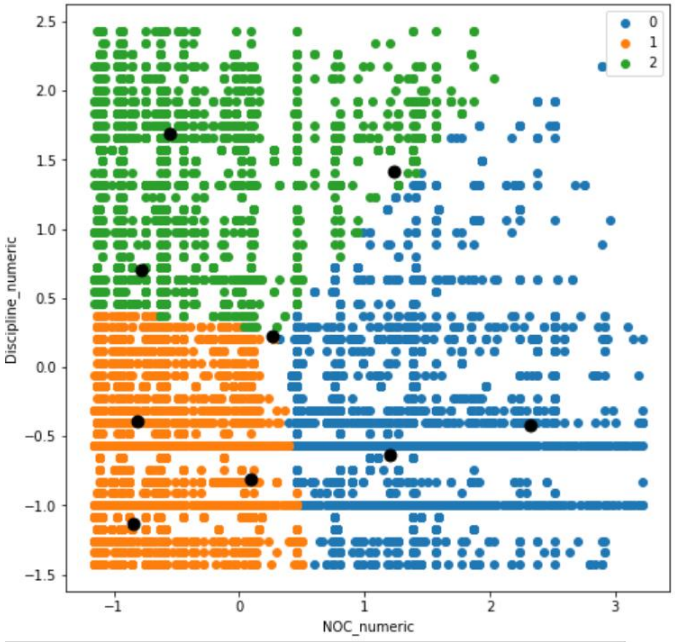
برای kmeans ابتدا باید تعداد بهینه خوشه ها را بدست آوریم که برای اینکار از روش Elbow و خطای SSE استفاده میکنیم. نمودار خطاها برای تعداد مختلف خوشه های kmeans به صورت زیر می باشد:



نمودار ۳۸ - نمودار خطای مدل kmeans در تعداد مختلف خوشه ها

همانطور که میبینیم مقدار بهینه k در تعداد خوشه ۳ رخ داده است. بنابراین تعداد خوشه را برابر با این مقدار میگذاریم و داده های نرمال شده را با آن خوشه بندی میکنیم.

بعد از خوشه بندی داده ها، نمودار خوشه های داده ها به صورت زیر می باشد:

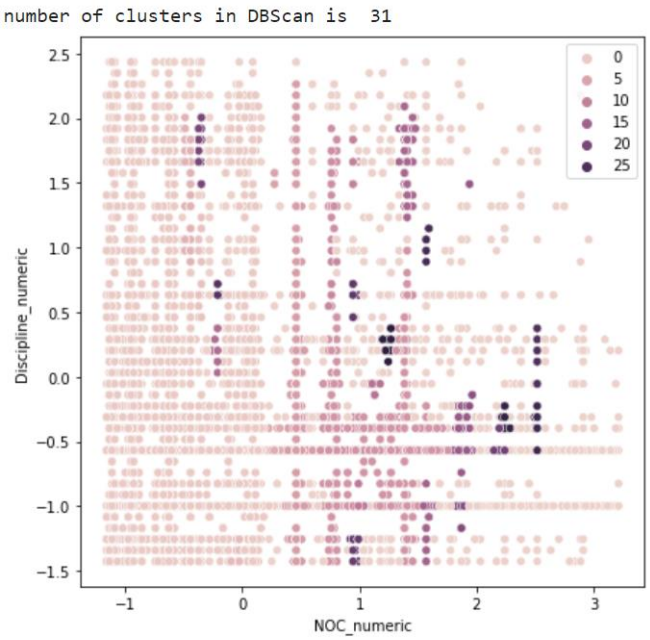


نمودار ۳۹ - نمودار داده های خوشه بندی شده با مدل kmeans با k = ۳

۴,۱,۲- الگوریتم DBScan

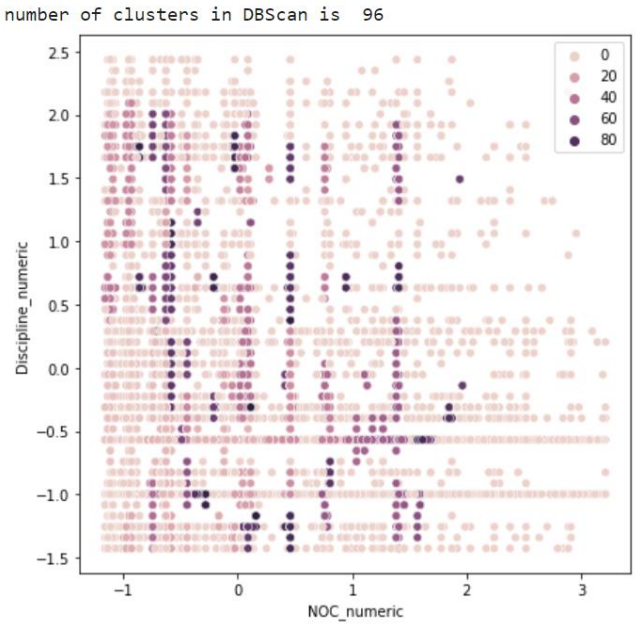
برای الگوریتم DBScan ما با مقادیر مختلف min_samples و eps امتحان کردیم.

مثلا برای eps=۲ و min_samples = ۲۰ خوشه ها به صورت زیر ایجاد شدند:



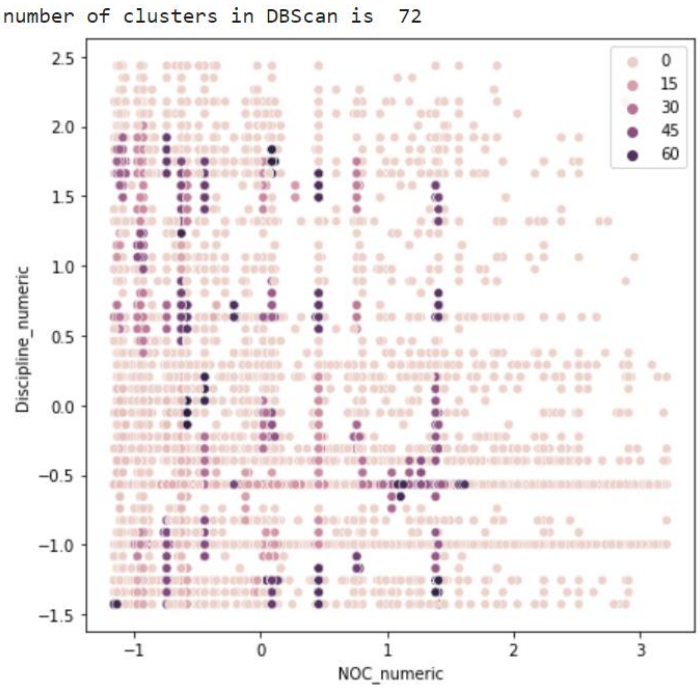
نمودار ۴۰ – نمودار خوشه بندی الگوریتم DBScan با min_samples=۲۰ و eps=۲

یا مثلا برای eps=۱ نیز خوشه بندی به صورت زیر است:



نمودار ۴۱ – نمودار خوشه بندی الگوریتم DBScan با min_samples=۲۰ و eps=۱

همچنین برای min_samples=۳۰ و eps=۱ خوشه بندی به صورت زیر می باشد:



نمودار ۴۲- نمودار خوشه بندی الگوریتم DBScan با min_samples=۳۰ و eps=۱

مشاهده میکنیم در حالتی که min_samples=۲۰ و eps =۲ است تعداد خوشه ها برابر با ۳۱ است و در حالتی که min_samples=۲۰ و eps=۱ می باشد تعداد خوشه ها ۷۱ یعنی دو برابر حالت eps=۲ است. پس نتیجه میگیریم با افزایش مقدار eps انتظار می‌رود تعداد خوشه ها کاهش یابد.

همچنین در حالتی که min_samples=۳۰ و eps=۱ است تعداد خوشه ها برابر با ۷۲ می باشد که یعنی تعداد خوشه ها نسبت به حالتی که min_samples=۲۰ و eps=۱ است یکی افزایش داشته.

۴,۱,۳- مقایسه دو الگوریتم Kmeans و DBScan

با مشاهده نمودار ۳۹ و نمودارهای ۴۰ تا ۴۲ متوجه میشویم که خوشه بندی kmeans نسبت به dbscan برای این داده ها بهتر عمل کرده و در نتیجه کارایی بیشتری دارد.

۴,۲- پیاده سازی الگوریتم های با نظارت

در این بخش ما از داده های خوشه بندی شده توسط kmeans در الگوریتم های بدون نظارت یعنی داده های ورزشکاران که براساس کشور و رشته ورزشی خوشه بندی شده اند میخواهیم برای قسمت الگوریتم های با نظارت استفاده کنیم. این داده ها براساس شباهتشان در ۳ خوشه قرار گرفته اند که در ادامه مدل ها را برای این داده ها آموزش میدهیم.

۴,۲,۱- مدل سازی شبکه عصبی

ابتدا لازم بود تا برای این بخش داده ها برچسب داده ها را به صورت categorical تبدیل کنیم. سپس داده ها را به ۳ دسته train، test و validation به صورت ۲۰ درصد داده test و ۲۰ درصد validation و باقی داده ها یعنی ۶۰ درصد داده های train را تشکیل میدهند.

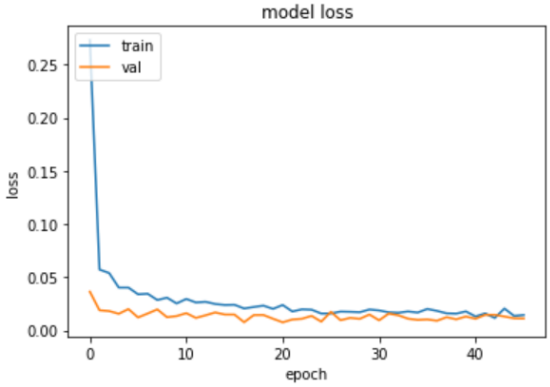
معماری شبکه ای که در نظر گرفتم به صورت زیر می باشد:

Layer (type)	Output Shape	Param #
dense_128 (Dense)	(None, 128)	384
dropout_1 (Dropout)	(None, 128)	0
dense_64 (Dense)	(None, 64)	8256
dropout_2 (Dropout)	(None, 64)	0
dense_32 (Dense)	(None, 32)	2080
dropout_3 (Dropout)	(None, 32)	0
dense_16 (Dense)	(None, 16)	528
dropout_4 (Dropout)	(None, 16)	0
dense_out (Dense)	(None, 3)	51

Total params: 11,299
Trainable params: 11,299
Non-trainable params: 0

تصویر ۱۶ - معماری شبکه عصبی

با در نظر گرفتن optimizer = Adam و learning_rate = ۰,۰۰۰۵ و loss='categorical_crossentropy' و در نظر گرفتن early_stopping با paitent = ۲۵ نمودار loss مربوط به داده های train و validation به صورت زیر می باشد:



نمودار ۴۳- نمودار loss شبکه عصبی ابتدایی

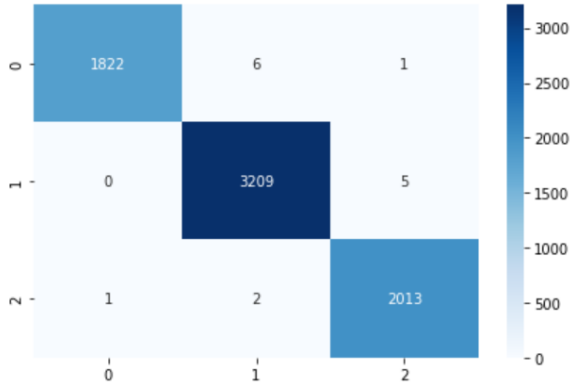
همانطور که مشاهده میکنید به دلیل استفاده از early_stop شبکه آموزش خود را در حدود epoch ۵۰ ام متوقف کرده تا overfit نشود.

و در نهایت روی داده های test به دقت زیر رسیدم:

```
69/69 [==============================] - 0s 3ms/step - loss: 0.0088 - accuracy: 0.9964
[0.008827632293105125, 0.9963735342025757]
```

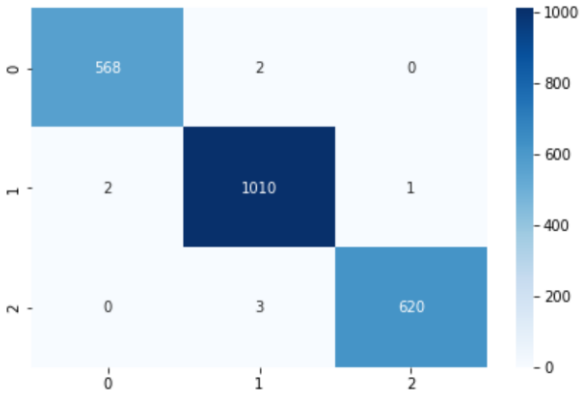
همچنین ماتریس confusion برای داده های تست و داده های آموزش به صورت زیر می باشد:

```
Accuracy Score on Train set : 0.9978750531236719
Train Confusion matrix
```



تصویر ۱۷ – ماتریس **confusion** برای داده های آموزش

```
Accuracy Score on Test set : 0.9963735267452403
Test Confusion matrix
```

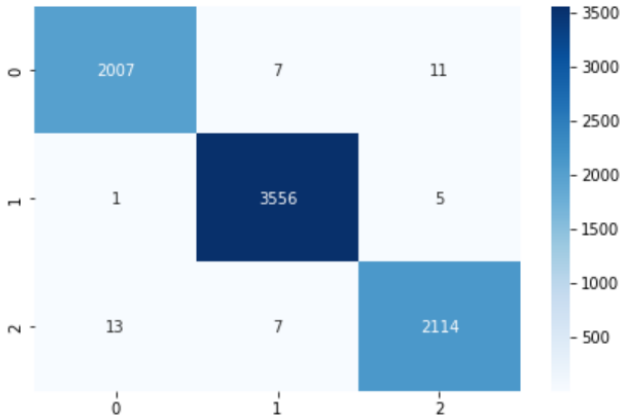


تصویر ۱۸ – ماتریس **confusion** برای داده های تست

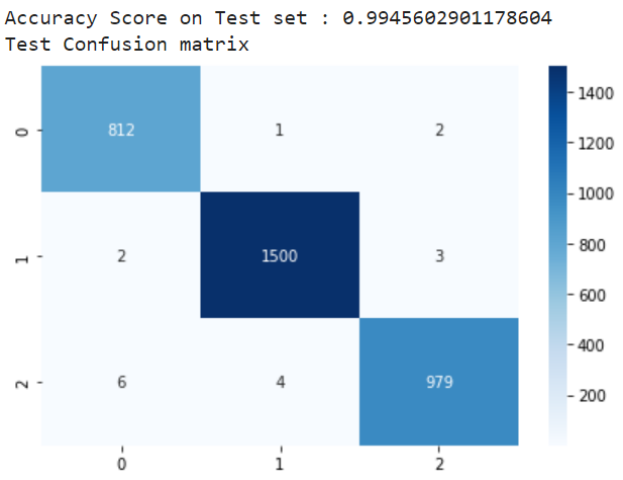
۴،۲،۲- الگوریتم طبقه بندی SVM

تلاش ۱: از مدل svm با کرنل linear استفاده کردم و دقت را اندازه گرفتم:

```
Accuracy Score on Train set : 0.994301256313949
Train Confusion matrix
```

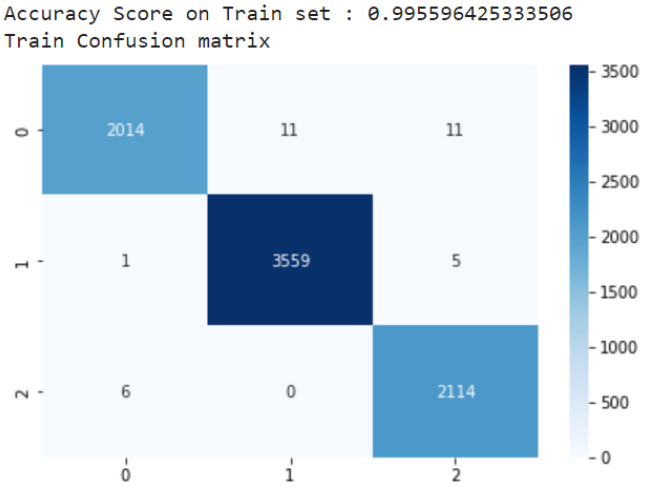


تصویر ۱۹ – ماتریس **confusion** برای داده های آموزش برای مدل **svm** با کرنل خطی

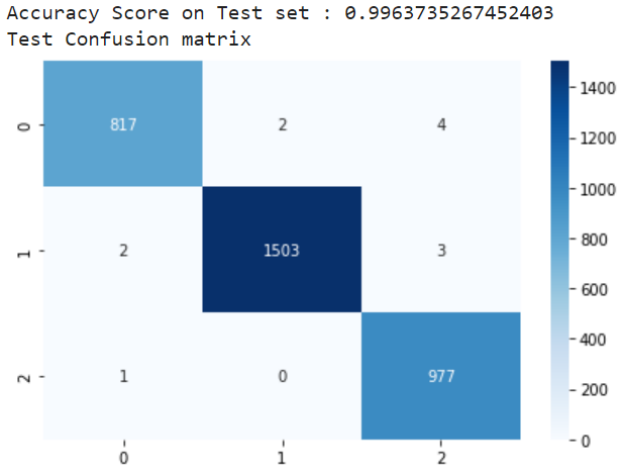


تصویر ۲۰ – ماتریس confusion برای داده های تست برای مدل svm با کرنل خطی

تلاش ۲: از مدل svm با کرنل rbf استفاده کردم و دقت را اندازه گرفتم:



تصویر ۲۱ – ماتریس confusion برای داده های آموزش برای مدل svm با کرنل rbf

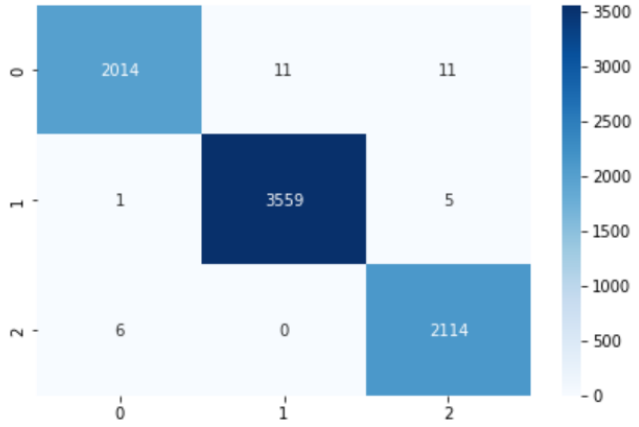


تصویر ۲۲ – ماتریس confusion برای داده های تست برای مدل svm با کرنل rbf

همانطور که در بالا مشاهده میکنیم دقت به مقدار ناچیز نسبت حالتی که با کرنل خطی بود افزایش داشته است.

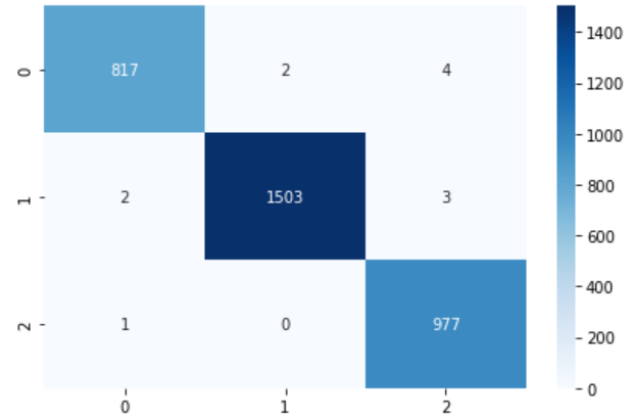
تلاش۳: در ۲ حالت قبل مقدار پارامتر gamma در حالت پیش فرض و با مقدار scale بود پس مقدار gamma را امتحان میکنم تا ببینم با تغییرات gamma چه خروجی خواهم داشت. پس مقدار gamma را از حالت scale به حالت auto تغییر دادیم و به دقت زیر رسیدیم:

```
Accuracy Score on Train set : 0.995596425333506
Train Confusion matrix
```



تصویر ۲۳ – ماتریس confusion برای داده های آموزش برای مدل svm با کرنل rbf و مقدار gamma = auto

```
Accuracy Score on Test set : 0.9963735267452403
Test Confusion matrix
```

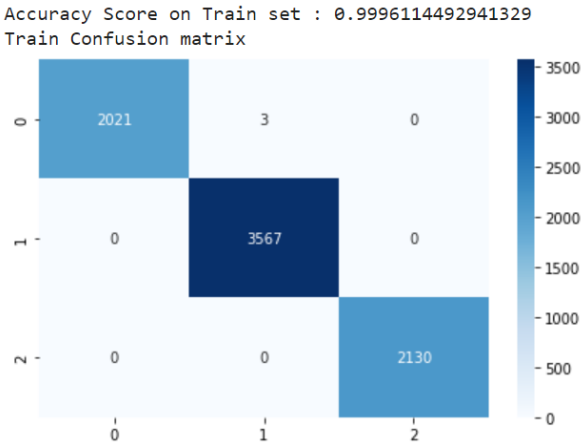


تصویر ۲۴ – ماتریس confusion برای داده های تست برای مدل svm با کرنل rbf و مقدار gamma = auto

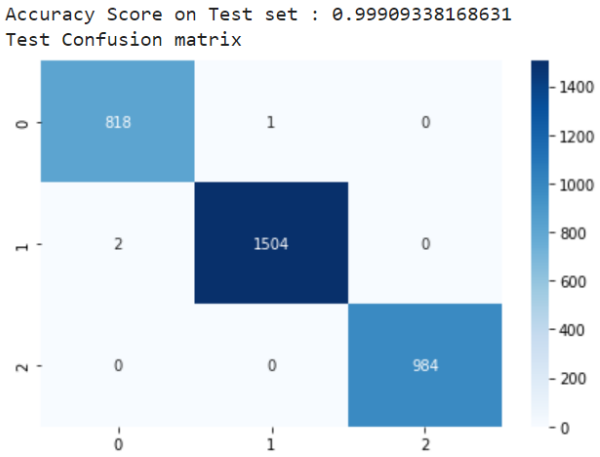
همانطور که میبینید نسبت به حالت قبل دقت تغییری نداشته است.

```
k-fold evaluate for svm model:  0.9952855847688123
```

تلاش ۴: مقدار gamma را اعداد ۵۰ و ۱۰۰۰ قرار دادم در نهایت در مقدار ۵۰ ، دقت مشابه بالا در حد چند هزارم بیشتر شد که در زیر ماتریس confusion و دقت مدل روی داده های آموزش و داده های تست را مشاهده میکنیم:



تصویر ۲۵- ماتریس confusion برای داده های آموزش برای مدل svm با کرنل rbf و مقدار gamma = ۵۰



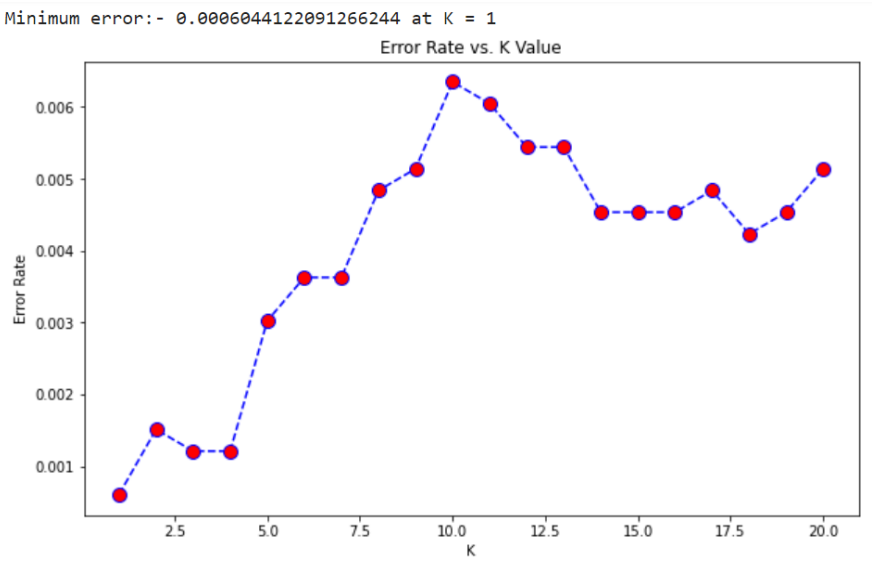
تصویر ۲۶ - ماتریس confusion برای داده های آزمون برای مدل svm با کرنل rbf و مقدار gamma = ۵۰

نکته: در حالتی که مقدار gamma برابر با auto می باشد، مدت زمان آموزش بسیار کمتر از حالتی است که به gamma مقدار ثابت می دهیم. بنابراین برای ما مقدار auto که دقت آن مشابه زمانی است که gamma برابر با ۵۰ هست، ارجحیت دارد.

۴,۲,۳- الگوریتم طبقه بندی KNN

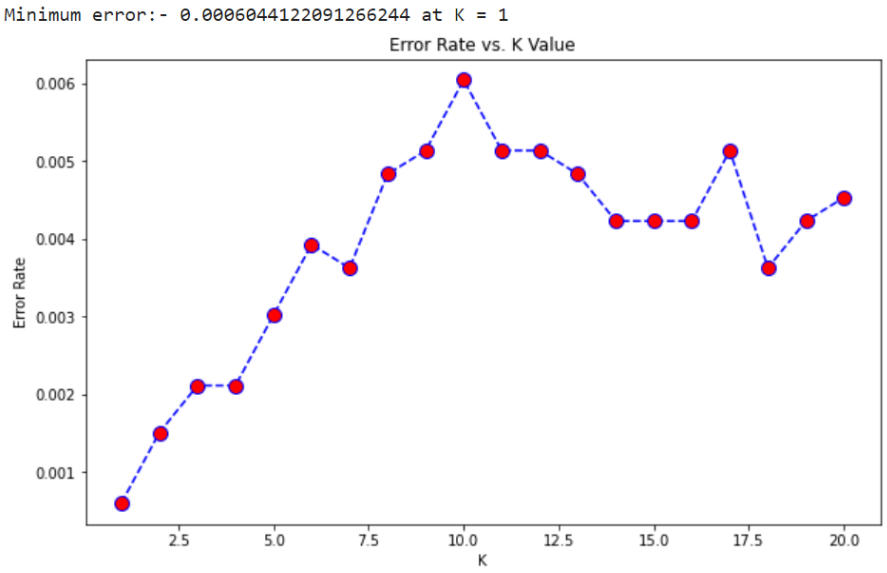
برای پیدا کردن مقدار بهینه تعداد همسایه ها، برای تعداد همسایه از ۱ تا ۲۰ مقدار خطا را محاسبه کردیم و نمودار خطا ها را رسم کردیم که به صورت زیر می باشد:

برای metric = ‘minkowski’ و p=۱ (یعنی فاصله منهتن):



نمودار ۴۴ - نمودار خطای مدل knn با متریک فاصله منهتن

برای ’metric = ‘minkowski و p=۲ (یعنی فاصله اقلیدسی):

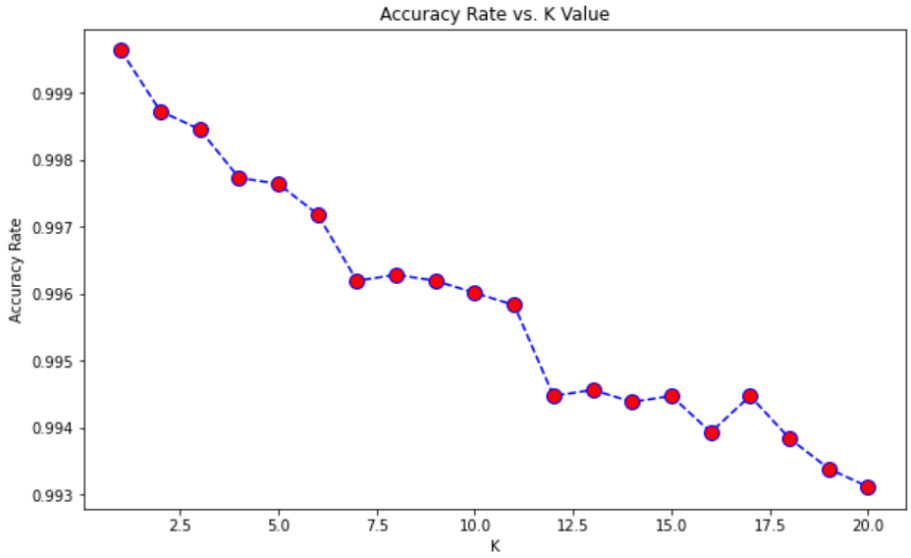


نمودار ۴۵ - نمودار خطای مدل knn با متریک فاصله اقلیدسی

پس از این ما مقدار دقت را براساس متریک ارزیابی k-fold برای تعداد همسایه ۱ تا ۲۰ محاسبه کردیم که نمودار آن به صورت زیر است:

برای ’metric = ‘minkowski و p=۱ (یعنی فاصله منهتن):

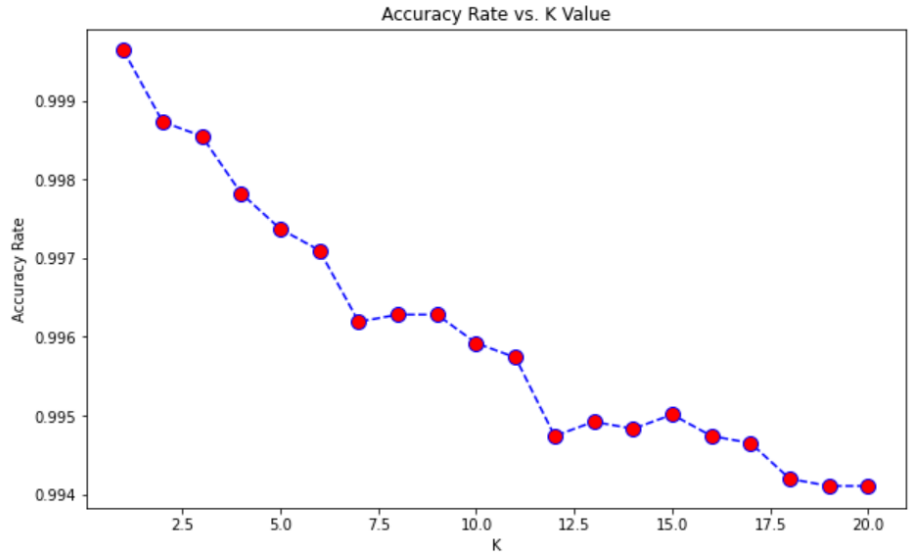
```
Max accuracy:- 0.9996373526745239 at K = 1
```



نمودار ۴۶ – نمودار دقت مدل knn با متریک فاصله منهتن

برای metric = ‘minkowski’ و p=۲ (یعنی فاصله اقلیدسی):

```
Max accuracy:- 0.9996373526745239 at K = 1
```



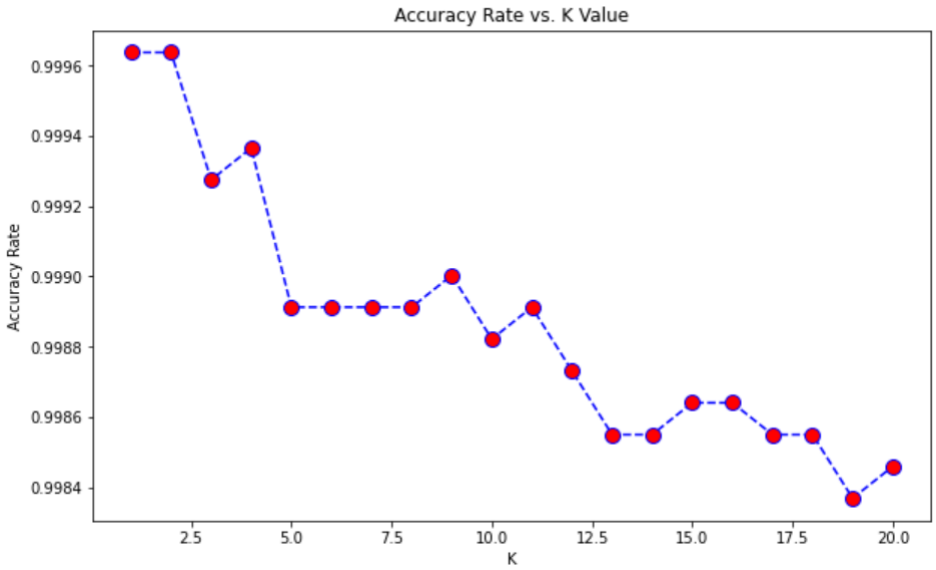
نمودار ۴۷ – نمودار دقت مدل knn با متریک فاصله اقلیدسی

همانطور که میبینیم تفاوتی بین میزان خطا و دقت در دو حالت فاصله اقلیدسی و منهتن نمی‌باشد.

تا اینجا که بررسی کردیم وزن همسایه ها یکسان بود. حال وزن همسایه ها براساس فاصله تغییر میکند که با این تغییر ما دوباره دقت را با k-fold بدست آوردیم:

برای metric = ‘minkowski’ و p=۱ (یعنی فاصله منهتن):

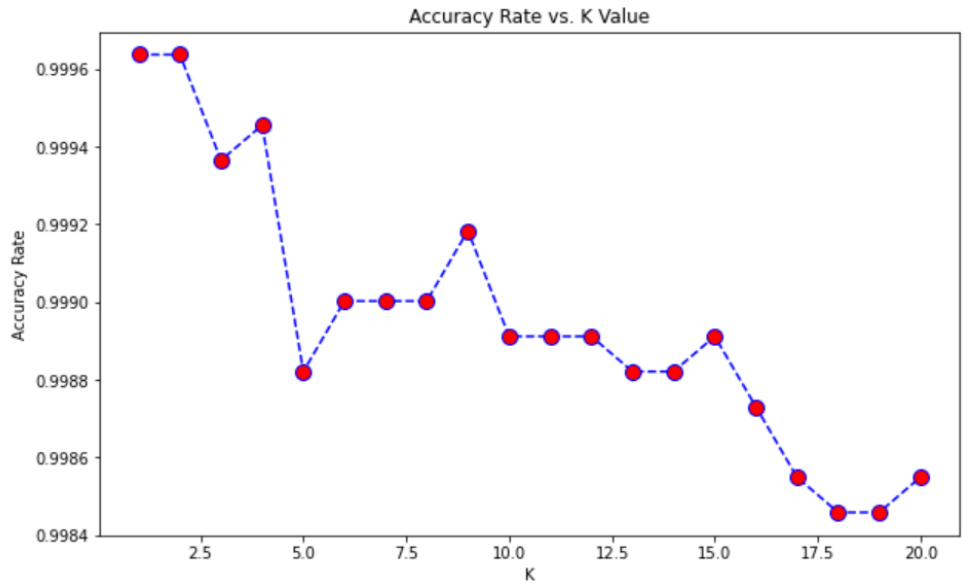
```
Max accuracy:- 0.9996373526745239 at K = 1
```



نمودار ۴۸ – نمودار دقت مدل knn با متریک فاصله منهتن و با وزن دهی به همسایه ها

برای metric = ‘minkowski’ و p=۲ (یعنی فاصله اقلیدسی):

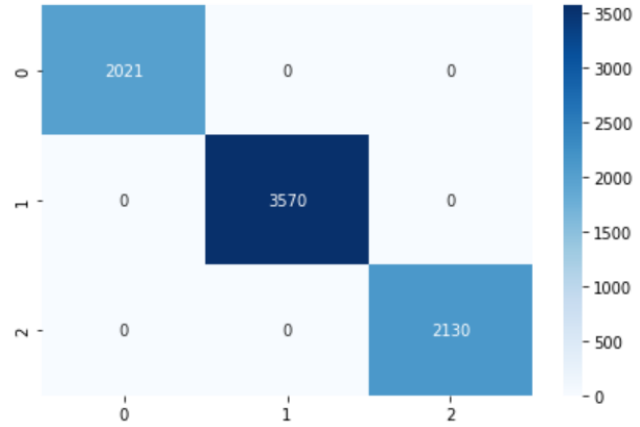
```
Max accuracy:- 0.9996373526745239 at K = 1
```



نمودار ۴۹ – نمودار دقت مدل knn با متریک فاصله اقلیدسی و وزن دهی به همسایه ها

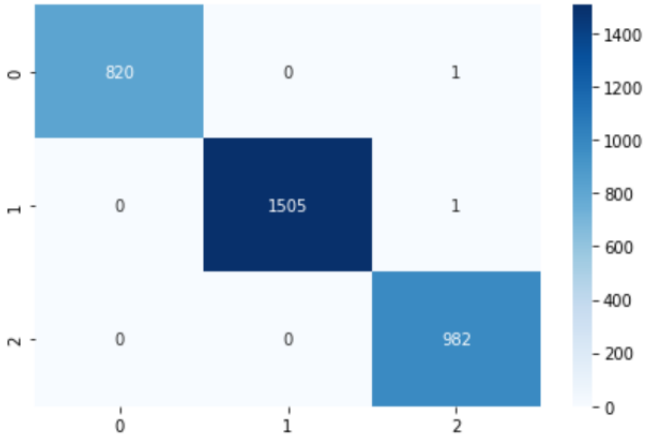
در نهایت با بررسی ها تصمیم گرفتیم که با تعداد همسایه ۲ و متریک فاصله اقلیدسی و همچنین دادن وزن به همسایه ها براساس میزان نزدیکی آنها به داده، مدل knn را آموزش دهیم که در نهایت این مدل روی داده های آموزش و تست دقت های زیر را دارد:

```
Accuracy Score on Train set : 1.0
Train Confusion matrix
```



تصویر ۲۷- ماتریس confusion برای داده های آموزش برای مدل نهایی knn

```
Accuracy Score on Test set : 0.9993955877908733
Test Confusion matrix
```



تصویر ۲۸ - ماتریس confusion برای داده های تست برای مدل نهایی knn

و با روش ارزیابی k-fold دقت به صورت زیر می باشد:

```
k-fold evaluate for knn model: 1.00
```

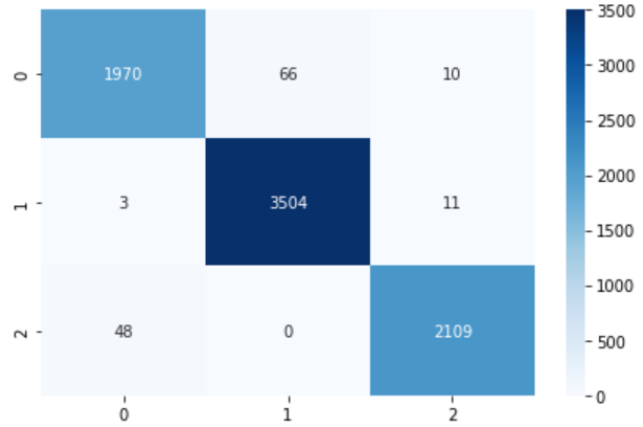
۴،۲،۴- الگوریتم طبقه بندی bayes

در این بخش ابتدا با استفاده از متد correlated_attrs که در بخش های قبل پیاده شده چک کردیم که آیا در دیتافریم athletes_for_model (همان دیتافریم که در بخش الگوریتم های بی نظارت برچسب زده شد) ویژگی های همبسته وجود دارد یا نه و از آنجایی که ویژگی همبسته ای وجود نداشت در نتیجه میتوانیم از naive Bayesian classifier استفاده کنیم.

بنابراین ما این مدل را ایجاد کردیم و روی داده ها آن را train کردیم.

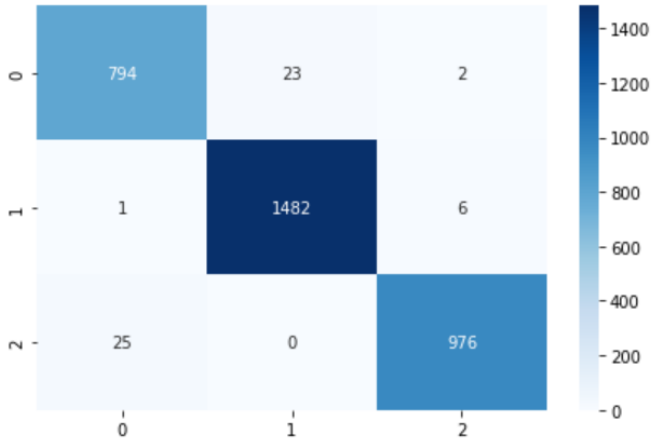
ماتریس confusion برای مدل bayes برای داده های تست و آموزش به صورت زیر می باشد:

```
Accuracy Score on Train set : 0.9821266675301127
Train Confusion matrix
```



تصویر ۲۹ - ماتریس confusion برای داده های آموزش برای مدل bayes

```
Accuracy Score on Test set : 0.9827742520398912
Test Confusion matrix
```



تصویر ۳۰ - ماتریس confusion برای داده های تست برای مدل bayes

پس از آن با روش ارزیابی k-fold دقت آن را محاسبه کردیم که نتیجه به صورت زیر شد:

```
k-fold evaluate for bayes model: 0.98
```

۴،۲،۵- الگوریتم طبقه بندی decision tree

تغییرات پارامتر ها:

۱. پارامتر criterion: این پارامتر متد انتخاب بهترین attribute برای جدا کردن داده ها را نشان میدهد که دو مقدار gini و entropy را دارد که gini برای روش gini impurity و entropy برای روش information gain می باشد. با مقدار دادن هر دو این روش ها درصد دقت مدل با روش k-fold یکسان و برابر با مقدار زیر شد:

```
k-fold evaluate for decision tree model: 1.00
```

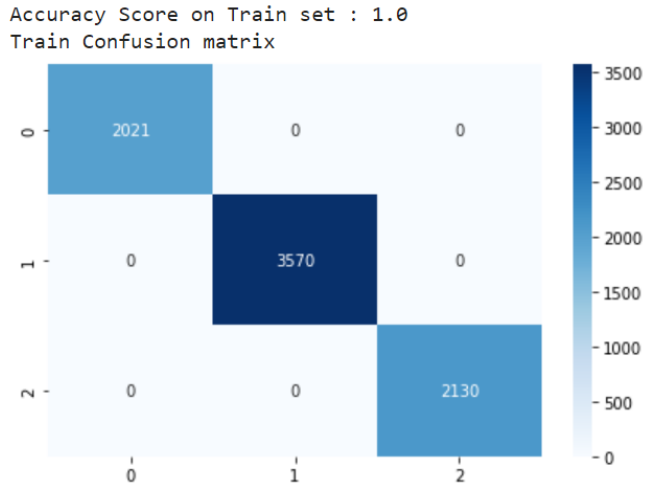
۲. پارامتر splitter: این پارامتر با دو مقدار best و random استراتژی مورد استفاده برای انتخاب تقسیم در هر گره را مشخص میکند. که مقدار best بهترین تقسیم رو برای هر گره انتخاب میکند و random بهترین انتخاب تصادفی را برای تقسیم در هر گره در نظر میگیرد. با هر دو مقدار best و random مقدار دقت ۱۰۰ درصد می باشد.

۳. Max_depth: بیشترین عمق مجاز برای درخت را تعیین میکند که مقدار پیش فرض ندارد و در نتیجه عمق درخت هر مقدار میتواند باشد. من مقدار عمق را ابتدا ۳ قرار دادم که دقت ۹۸ درصد شد، سپس ۴ قرار دادم و دقت ۹۹ درصد شد، پس از آن برای عمق بیشینه برابر و بیشتر از ۵ دقت برابر با ۱۰۰ درصد شد.

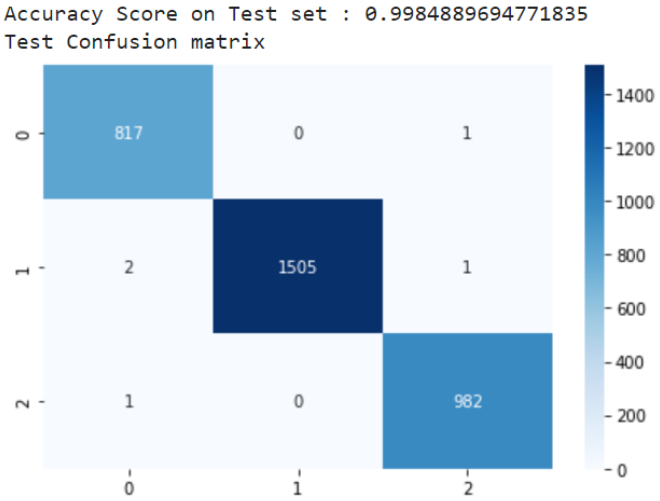
۴. ccp_alpha: هرس پیچیدگی حداقل هزینه، به صورت بازگشتی گره با ضعیفترین پیوند را انتخاب و هرس میکند. ضعیفترین پیوند با آلفای موثر مشخص میشود، جایی که گره های با آلفای کمتر از ccp_alpha هرس میشوند. مقدار پیش فرض این متغیر صفر است به این معنی که به صورت پیش فرض درخت هرس نمیشود.

۴,۲,۵,۱- درخت تصمیم بدون هرس

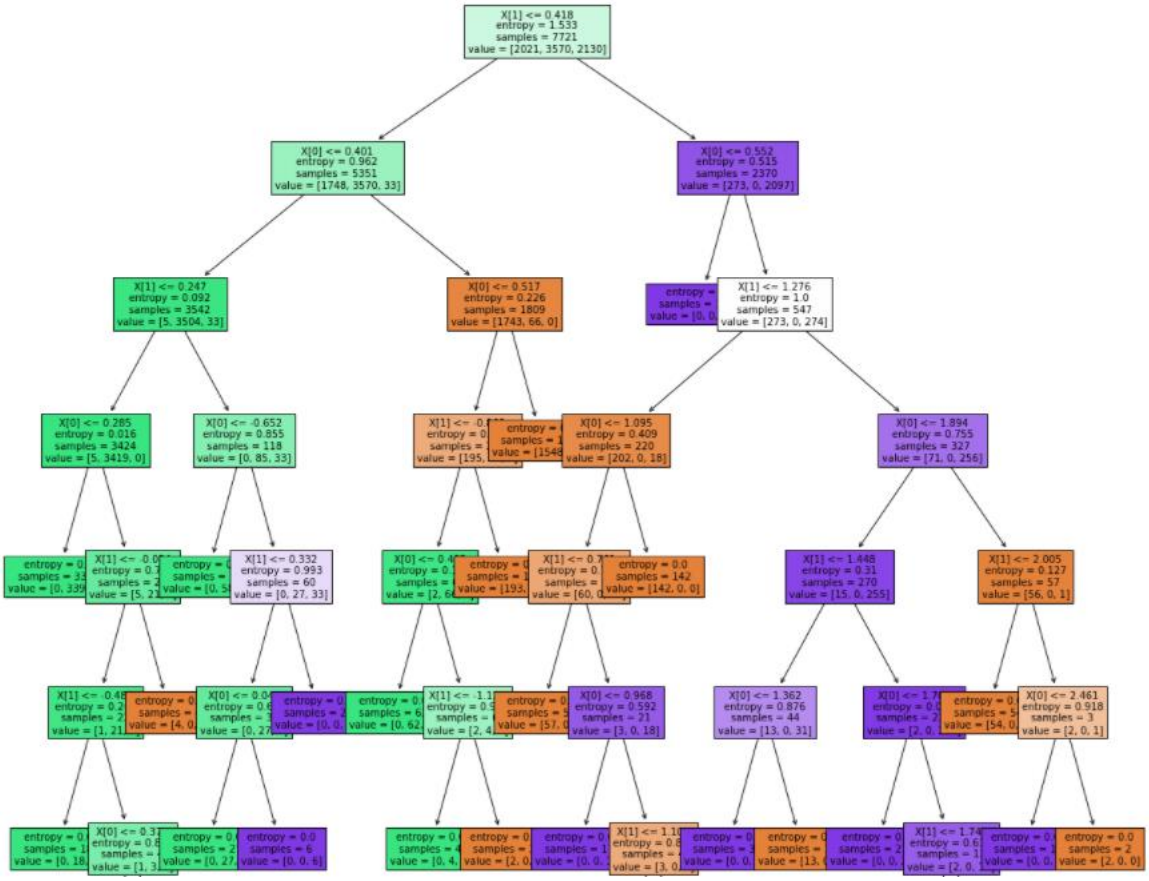
ماتریس confusion برای داده های آموزش و تست برای مدل درخت تصمیم به صورت زیر می باشد:



تصویر ۳۱ - ماتریس confusion برای داده های آموزش برای مدل درخت تصمیم با criterion = "entropy" و splitter = "best"



تصویر ۳۲ - ماتریس confusion برای داده های تست برای مدل درخت تصمیم با criterion = "entropy" و splitter = "best"



تصویر ۳۳ - شمایل درخت تصمیم با criterion = "entropy" و splitter = "random"

همچنین دقت مدل با روش k-fold به صورت زیر می باشد:

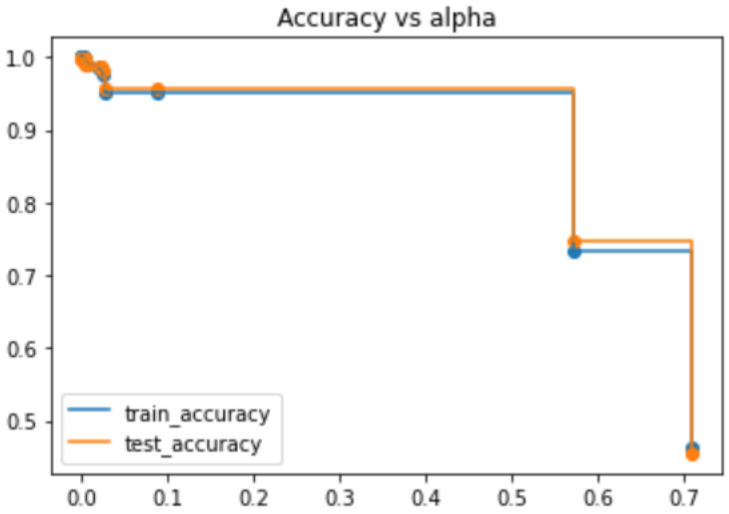
```
k-fold evaluate for decision tree model: 1.00
```

۴,۲,۵,۲- درخت تصمیم هرس شده

در بین پارامتر های معرفی شده برای درخت تصمیم پارامتری به نام ccp_alpha معرفی کردیم که برای هرس درخت کاربرد دارد و مورد استفاده قرار میگیرد.

برای هرس درخت تصمیم ابتدا باید مقدار بهینه برای ccp_alpha را پیدا کنیم. ابتدا برای ccp_alpha های مختلف مدل را train می کنیم و تمام مدل ها را در آرایه قرار میدهیم. سپس برای همه مدل ها داده های آموزش و تست را پیش بینی میکنیم و سپس با استفاده از برچسب واقعی و برچسب پیش بینی شده دقت را برای هر مدل محاسبه میکنیم.

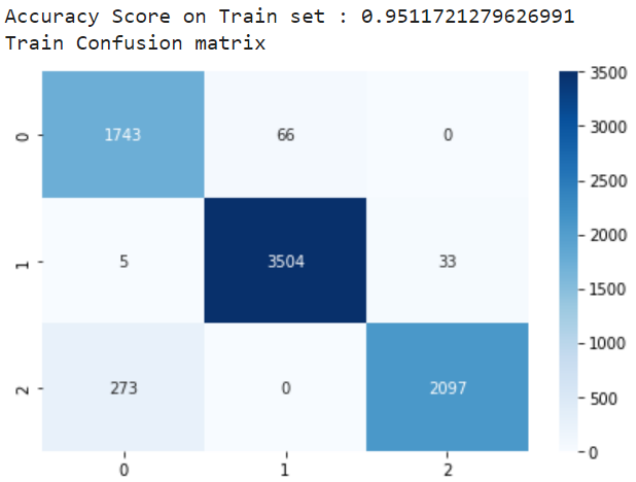
جدول زیر دقت داده های آموزش و تست براساس مقدار alpha می باشد:



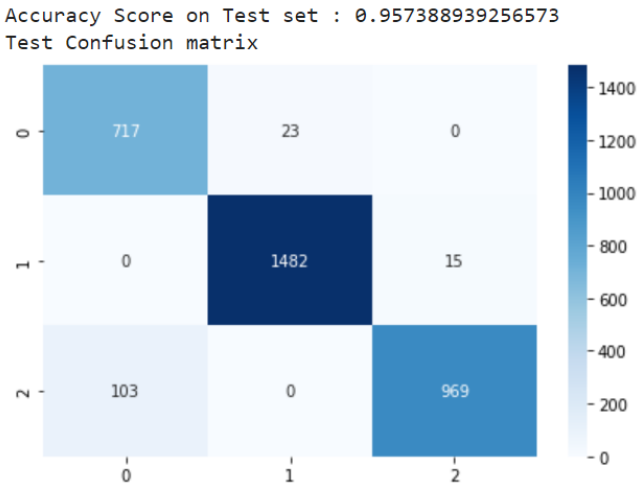
تصویر ۳۴ – دقت داده های آموزش و تست برای مدل ها براساس مقدار alpha

بر اساس این جدول تصمیم گرفتیم تا مقدار ccp_alpha را برابر با ۰.۵ قرار دهیم.

در نتیجه ماتریس confusion مدل هرس شده با criterion = “entropy” و splitter = “best” و ccp_alpha = ۰.۵ به صورت زیر می باشد:



تصویر ۳۵ – ماتریس confusion برای داده های آموزش برای مدل درخت تصمیم با ccp_alpha = ۰.۵



تصویر ۳۶ – ماتریس confusion برای داده های آموزش برای مدل درخت تصمیم با ccp_alpha = ۰.۵

همچنین دقت مدل هرس شده با روش k-fold به صورت زیر می باشد:

```
k-fold evaluate for decision tree model: 0.95
```

همانطور که میبینیم نسب به حالت هرس نشده دقت کمی کاهش پیدا کرد.

نکته: در ابتدا که داشتیم پارامتر های الگوریتم را معرفی میکردیم گفتیم که splitter با مقدار best و random دقت یکسان و برابر ۱۰۰ داشت، اما این حالت برای درخت بدون هرس بود و برای درخت هرس شده متوجه شدیم که درصورتی که مقادیر دیگر پارامتر ها مشابه بالا باشد، آنگاه دقت در حالتی که splitter برابر با best است برابر با ۹۵ درصدو در حالت random برابر با ۷۰ درصد می باشد که همانطور که میبینیم تفاوت بسیار زیاد است. لذا در اینجا مقدار best برای پارامتر splitter بسیار بهتر است و دقت بهترین نتیجه میدهد.

۴,۲,۶- روش ensemble

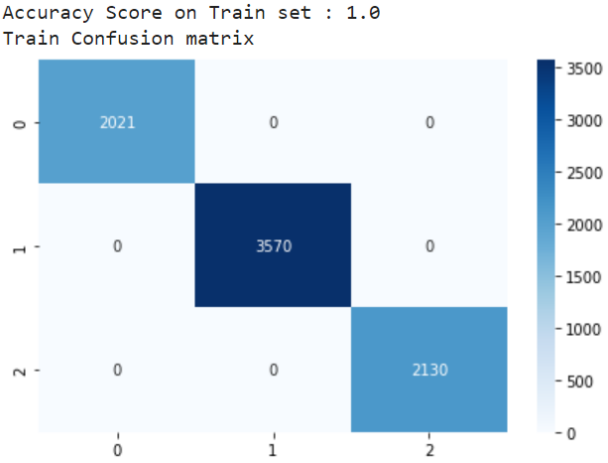
۴,۲,۶,۱ – استفاده از VotingClassifier

در اینجا ما سعی کردیم بهترین حالات الگوریتم های طبقه بندی که در بخش های قبل به آنها رسیدیم را استفاده کنیم. بنابراین در این مرحله ما از مدل svm و knn و decision tree با دقت ۱۰۰٪ استفاده کردیم. همچنین از دسته بند RandomForestClassifier نیز استفاده کردیم که دقت آن نیز ۱۰۰٪ می باشد.

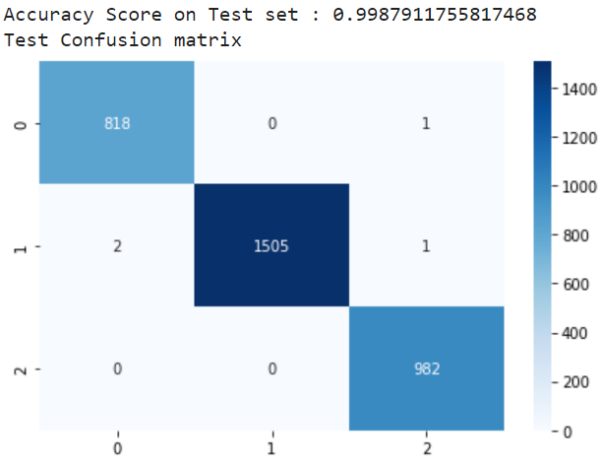
وقتی از این ۴ الگوریتم در ensemble استفاده کردیم، دقت هر یک به صورت زیر شد:

```
Accuracy of SVM: 1.00
Accuracy of KNN: 1.00
Accuracy of Decision Tree: 1.00
Accuracy of Random Forest: 1.00
Accuracy of Ensemble: 1.00
```

بنابراین ماتریس confusion و دقت برای داده های آموزش و آزمون برای مدل ensemble به صورت زیر می باشد:



تصویر ۳۷- ماتریس confusion برای داده های آموزش برای مدل ensemble



تصویر ۳۸ – ماتریس confusion برای داده های آزمون برای مدل ensemble

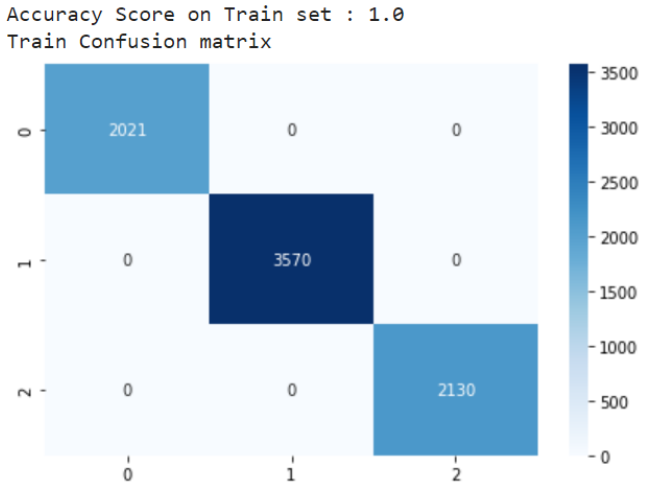
۴,۲,۶,۲- استفاده از ExtraTreesClassifier

این کلاس تعدادی درخت تصادفی موسوم به درخت های اضافی را روی مجموعه داده آموزش fit میکند و از روش میانگین گیری برای بهبود دقت و کنترل overfit استفاده میکند.

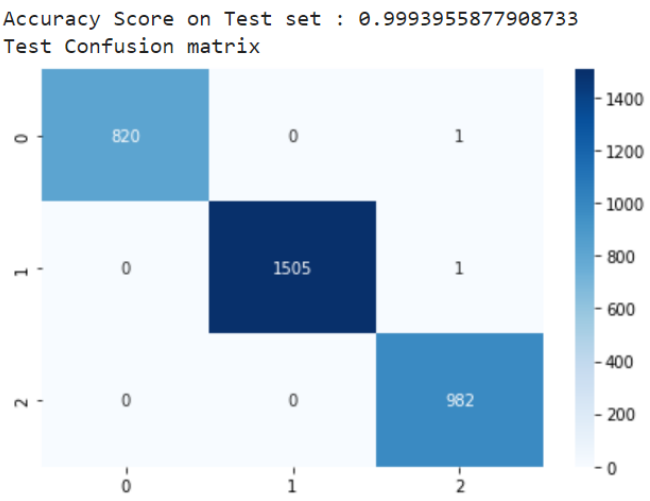
برای این حالت من تعداد estimator ها را (پارامتر n_stimators) را ۱۰ قرار دادم. همچنین پارامتر criterion که روش پیدا کردن بهترین ویژگی برای split را نشان میدهد برابر با entropy قرار دادم، که در نهایت با تعیین این پارامتر ها دقت مدل ۶۳٪ شد.

```
k-fold evaluate for extra-trees classifier model: 1.00
```

ماتریس confusion برای داده های آموزش و آزمون برای مدل ensemble از نوع ExtraTrees به صورت زیر می باشد:



تصویر ۳۹- ماتریس confusion برای داده های آموزش برای مدل ExtraTreesClassifier



تصویر ۴۰- ماتریس confusion برای داده های آزمون برای مدل ExtraTreesClassifier

بخش پنجم: الگوهای پرتکرار و قوانین انجمنی

۵,۱- الگوهای پرتکرار

ما برای پیدا کردن الگوهای پرتکرار ابتدا ستون های عددی که نگاشتی از ستون های رشته ای هستند را حذف کرده ایم چون ستون های رشته ای متناظر با آنها را در دیتافریم ها داریم.

پس از آن با استفاده از TransactionEncoder داده های داخل دیتافریم را به فرم encode شده درآوردیم و پس از آن الگوریتم apriori را روی دیتافریم مربوطه اعمال کردیم.

نکته- با توجه به اینکه در دیتاست ما داده های ما حالت گزارش دارند و معمولا اکثر سطر ها unique هستند، بنابراین به طور کلی الگوی پرتکرار با support بالا در آن وجود ندارد، با این حال یکسری قوانین را بدست آوردیم که در ادامه آنها را مشاهده میکنیم.

دیتافریم تیم ها که شامل نام تیم، رشته ورزشی، نام کشور و نوع رویداد است:

در این دیتافریم سطر هایی که مربوط به کشور ژاپن و آمریکا هستند از همه بیشترند. پس به عنوان مثال تصمیم گرفتیم برای تیم های کشور ژاپن و آمریکا الگوی پرتکرار پیدا کنیم:

```
find_frequent_patterns(teams_for_japan, 0.1)
```

	support	itemsets
0	0.106383	(Fencing)
1	1.000000	(Japan)
2	0.191489	(Men)
3	0.148936	(Swimming)
4	0.191489	(Women)
5	0.106383	(Japan, Fencing)
6	0.191489	(Men, Japan)
7	0.148936	(Japan, Swimming)
8	0.191489	(Japan, Women)

تصویر ۴۱- الگوهای پرتکرار مربوط به تیم های کشور ژاپن با minsup = ۰,۱

همانطور که میبینیم تمام itemset ها به غیر از (Japan) مقدار support کمتر از ۰,۲ دارند که بسیار پایین است.

```
find_frequent_patterns(teams_for_usa, 0.2)
```

	support	itemsets
0	1.0	(United States)
1	1.0	(United States of America)
2	1.0	(United States, United States of America)

تصویر ۴۲ - الگوهای پرتکرار مربوط به تیم های کشور آمریکا با minsup = ۰,۲

حال الگوی پرتکرار برای تیم های ژاپن و آمریکا (با هم) میخواهیم بدست آوریم:

```
find_frequent_patterns(teams_for_japan_usa, 0.2)
```

	support	itemsets
0	0.522222	(Japan)
1	0.477778	(United States)
2	0.477778	(United States of America)
3	0.477778	(United States, United States of America)

تصویر ۴۳ - الگوی پرتکرار مربوط به تیم های کشورهای آمریکا و ژاپن با minsup = ۰,۲

همانطور که میبینیم از ترکیب تیم های کشور های ژاپن و آمریکا الگوی پرتکرار جدیدی مشاهده نشد.

دیتافریم مربیان شامل نام مربی، نام کشور، رشته ورزشی و نوع رویداد می باشد:

ابتدا نام مربی را از دیتافریم حذف میکنیم چون مقدار آن منحصر به فرد است و کمکی در پیدا کردن الگوی پرتکرار به ما نمیکند. کشور های ژاپن و اسپانیا بیشترین تعداد مربی را دارند در نتیجه تصمیم گرفتیم تا برای این دو کشور الگوهای پرتکرار را پیدا کنیم:

```
find_frequent_patterns(coaches_for_japan,0.4)
```

	support	itemsets
0	1.000000	(Japan)
1	0.428571	(Men)
2	0.400000	(Women)
3	0.428571	(Men, Japan)
4	0.400000	(Japan, Women)

تصویر ۴۴ - الگوهای پرتکرار مربیان کشور ژاپن با min_sup = ۰,۴

```
find_frequent_patterns(coaches_for_spain, 0.3)
```

	support	itemsets
0	0.357143	(Basketball)
1	0.357143	(Men)
2	1.000000	(Spain)
3	0.357143	(Women)
4	0.357143	(Basketball, Spain)
5	0.357143	(Men, Spain)
6	0.357143	(Women, Spain)

تصویر ۴۵ - الگوهای پرتکرار مربیان کشور اسپانیا با min_sup = ۰,۴

حال میخواهیم از الحاق داده های مربیان ژاپن و اسپانیا الگوهای پرتکرار استخراج کنیم:

```
find_frequent_patterns(coaches_for_japan_spain,0.3)
```

	support	itemsets
0	0.555556	(Japan)
1	0.396825	(Men)
2	0.444444	(Spain)
3	0.380952	(Women)

تصویر ۴۶ - الگوهای پرتکرار مربیان کشور های ژاپن و اسپانیا با minsup = ۰,۳

همچنین برای یکی از کشور هایی که تعداد مربیان آن کم است نیز میخواهیم الگوهای پرتکرار را پیدا کنیم، در اینجا کشور فرانسه به تعداد ۱۰ مربی دارد که میخواهیم برای آن الگوهای پرتکرار را پیدا کنیم:

	support	itemsets
0	0.4	(Basketball)
1	0.2	(Duet)
2	1.0	(France)
3	0.2	(Handball)
4	0.4	(Men)
5	0.4	(Women)
6	0.4	(France, Basketball)
7	0.2	(Women, Basketball)
8	0.2	(France, Duet)
9	0.2	(France, Handball)
10	0.4	(France, Men)
11	0.4	(France, Women)
12	0.2	(France, Women, Basketball)

تصویر ۴۷- الگوهای پرتکرار مربیان کشور فرانسه با minsup = ۰,۲

فایل ورزشکاران شامل نام ورزشکار، نام کشور، رشته ورزشی می باشد:

در ابتدای کار نام ورزشکاران را از دیتافریم حذف میکنیم چون مقادیر آن منحصر به فرد است و در پیدا کردن الگوهای پرتکرار کمک نمیکند.

در این فایل کشور های آمریکا، ژاپن و استرالیا بیشترین تعداد ورزشکار را دارند پس برای این ورزشکاران این کشور ها الگوهای پرتکرار را پیدا میکنیم:

```
find_frequent_patterns(athletes_for_fp_usa, 0.2)
```

	support	itemsets
0	0.234146	(Athletics)
1	1.000000	(United States of America)
2	0.234146	(Athletics, United States of America)

تصویر ۴۸ - الگوهای پرتکرار ورزشکاران کشور آمریکا با minsup = ۰,۲

```
find_frequent_patterns(athletes_for_fp_japan, 0.1)
```

	support	itemsets
0	0.119454	(Athletics)
1	1.000000	(Japan)
2	0.119454	(Japan, Athletics)

تصویر ۴۹ - الگوهای پرتکرار ورزشکاران کشورژاپن با minsup = ۰,۱

```
find_frequent_patterns(athletes_for_fp_australia, 0.1)
```

	support	itemsets
0	0.138298	(Athletics)
1	1.000000	(Australia)
2	0.138298	(Australia, Athletics)

تصویر ۵۰ - الگوهای پرتکرار ورزشکاران کشور استرالیا با minsup=۰,۱

همانطور که میبینیم برای این ها نیز الگوهای پرتکرار با support بالا پیدا نشده است.

۵,۲- قوانین انجمنی

حال برای الگوهای پرتکراری که در بخش قبل پیدا کردیم، قوانین انجمنی را استخراج میکنیم. در این بخش از معیار همبستگی lift برای ارزیابی قوانین استفاده کردیم. لازم به ذکراست که در اینجا کمترین مقدار اطمینان ۰,۶ در نظر گرفته شده است که با توجه به این قوانین انجمنی به صورت زیر می باشد:

```
find_association_rules(fp_of_teams_in_japan)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Fencing)	(Japan)	0.106383	1.0	0.106383	1.0	1.0	0.0	inf
1	(Men)	(Japan)	0.191489	1.0	0.191489	1.0	1.0	0.0	inf
2	(Swimming)	(Japan)	0.148936	1.0	0.148936	1.0	1.0	0.0	inf
3	(Women)	(Japan)	0.191489	1.0	0.191489	1.0	1.0	0.0	inf

تصویر ۵۱- قوانین انجمنی برای تیم های کشور ژاپن با minsup = ۰.۱ و minconf = ۰.۶

```
find_association_rules(fp_of_teams_in_usa)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(United States of America)	(United States)	1.0	1.0	1.0	1.0	1.0	0.0	inf
1	(United States)	(United States of America)	1.0	1.0	1.0	1.0	1.0	0.0	inf

تصویر ۵۲ - قوانین انجمنی برای تیم های کشور آمریکا با minsup = ۰.۲ و minconf = ۰.۶

برای قوانین دو جدول بالا معیار همبستگی lift برابر ۱ می باشد در نتیجه میتوان گفت که این قوانین از لحاظ آماری نمیتواند در دنیای واقعی استفاده شوند.

```
find_association_rules(fp_of_teams_in_japan_usa)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(United States of America)	(United States)	0.477778	0.477778	0.477778	1.0	2.093023	0.249506	inf
1	(United States)	(United States of America)	0.477778	0.477778	0.477778	1.0	2.093023	0.249506	inf

تصویر ۵۳ - قوانین انجمنی برای تیم های کشور ژاپن و آمریکا با minsup = ۰.۲ و minconf = ۰.۶

در تصویر ۵۳ مقدار معیار همبستگی lift برای قوانین بیشتر از ۱ می باشد، در نتیجه میتوان گفت که این قوانین، قوی هستند و همبستگی مثبت وجود دارد.

برای جداول بعد نیز همانطور که مشاهده میکنید مقدار lift برابر ۱ می باشد که نشان میدهد که این قوانین، قوی نیستند و در دنیای واقعی کاربرد ندارند.

```
find_association_rules(fp_of_coaches_in_japan)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Men)	(Japan)	0.428571	1.0	0.428571	1.0	1.0	0.0	inf
1	(Women)	(Japan)	0.400000	1.0	0.400000	1.0	1.0	0.0	inf

تصویر ۵۴ - قوانین انجمنی برای مربیان کشور ژاپن با minsup = ۰.۴ و minconf = ۰.۶

```
find_association_rules(fp_of_coaches_in_spain)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Basketball)	(Spain)	0.357143	1.0	0.357143	1.0	1.0	0.0	inf
1	(Men)	(Spain)	0.357143	1.0	0.357143	1.0	1.0	0.0	inf
2	(Women)	(Spain)	0.357143	1.0	0.357143	1.0	1.0	0.0	inf

تصویر ۵۵- قوانین انجمنی برای مربیان کشور اسپانیا با minsup = ۰,۴ و minconf = ۰,۶

```
find_association_rules(fp_of_coaches_in_japan_spain)
```

antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction

تصویر ۵۶ – قوانین انجمنی برای مربیان کشور ژاپن و اسپانیا با minsup = ۰,۳ و minconf = ۰,۶

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Basketball)	(France)	0.4	1.0	0.4	1.0	1.0	0.0	inf
1	(Duet)	(France)	0.2	1.0	0.2	1.0	1.0	0.0	inf
2	(Handball)	(France)	0.2	1.0	0.2	1.0	1.0	0.0	inf
3	(Men)	(France)	0.4	1.0	0.4	1.0	1.0	0.0	inf
4	(Women)	(France)	0.4	1.0	0.4	1.0	1.0	0.0	inf

تصویر ۵۷ – قوانین انجمنی برای مربیان کشور فرانسه با minsup = ۰,۲ و minconf = ۰,۶

```
find_association_rules(fp_of_athletes_in_usa)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Athletics)	(United States of America)	0.234146	1.0	0.234146	1.0	1.0	0.0	inf

تصویر ۵۸ – قوانین انجمنی برای ورزشکاران کشور آمریکا با minsup = ۰,۲ و minconf = ۰,۶

```
find_association_rules(fp_of_athletes_in_japan)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Athletics)	(Japan)	0.119454	1.0	0.119454	1.0	1.0	0.0	inf

تصویر ۵۹ –قوانین انجمنی برای ورزشکاران کشور ژاپن با minsup = ۰,۱ و minconf = ۰,۶

```
find_association_rules(fp_of_athletes_in_australia)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Athletics)	(Australia)	0.138298	1.0	0.138298	1.0	1.0	0.0	inf

تصویر ۶۰ – قوانین انجمنی برای ورزشکاران کشور استرالیا با minsup = ۰,۱ و minconf = ۰,۶

نتیجه گیری

ما در این پروژه سعی کردیم تا فاز های مختلف داده کاوی را روی مجموعه داده خود انجام دهیم. ما در ابتدا در این پروژه از داده های موجود در دیتافریم مربیان برای الگوریتم های بانظارت استفاده کردیم و گفتیم ستون event برچسب و ستون کشور و رشته ورزشی ویژگی های داده باشند، اما بعد از آموزش و آزمون داده ها دیدیم که دقت مدل ها بسیار پایین می باشد.

بنابراین در قدم بعدی داده هایی که توسط مدل kmeans که خوشه بندی مناسبی داشت یعنی داده های دیتافریم ورزشکاران که بر اساس مقدار ستون های کشور و رشته ورزشی خوشه بندی شده بوده اند استفاده کردیم و نتیجه این شد که دقت مدل ها بالا رفت. درنتیجه متوجه شدیم که مشکل از مدل ها نمی باشد بلکه مشکل از داده هایی بود که استفاده کرده بودیم و ستونی که به اشتباه برچسب در نظر گرفته بودیم. البته کم بودن تعداد داده های دیتافریم مربیان به نسبت دیتافریم ورزشکاران نیز تاثیر زیادی در دقت داده ها داشت به خصوص تاثیر تعداد داده ها در مدل شبکه عصبی که برای آموزش مناسب به داده های زیادی نیاز دارد به وضوح مشاهده شد.

منابع و مراجع

[۱] https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn

[۲] https://towardsdatascience.com/bbn-bayesian-belief-networks-how-to-build-them-effectively-in-python-٦b٧f٩٣٤٣٥bba

[۳] https://mljar.com/blog/visualize-decision-tree/

[٤] https://scikit-learn.org/stable/modules/tree.html

[٥] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier

[٦] https://scikit-learn.org/stable/auto_examples/tree/plot_cost_complexity_pruning.html

[٧] https://www.analyticsvidhya.com/blog/٢٠٢٠/١٠/cost-complexity-pruning-decision-trees/

[٨] https://www.tensorflow.org/tutorials/keras/classification

[٩] https://scikit-learn.org/stable/modules/ensemble.html

[١٠] https://www.kaggle.com/faressayah/ensemble-ml-algorithms-bagging-boosting-voting

[١١] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html

[١٢] https://towardsdatascience.com/k-means-vs-dbscan-clustering-٤٩f٨e٦٢٧de٢٧

[١٣] https://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html