

**Tehran Polytechnic University**

**Computer Engineering Department**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Data Mining**

**Assignment Two**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Name: Mohammad Hossein Badiei**

**Student ID: 9531701**

**Majors: Artificial Intelligence and Robotics (Amirkabir) | Electrical Engineering (Tehran)**

**Instructor: Dr. Ehsan Nazerafard**

**Spring 2021**

**پاسخ سوال 1**

در این سوال خواسته شده که هر یک از مفاهیمِ موجود در سوال را تعریف کنیم.

1. Unsupervised Learning

یادگیریِ بدونِ نظارت یکی از روش‌های یادگیری است که بر روی داده‌های بدونِ برچسب اعمال می‌شود. در واقع در این نوع یادگیری آموزش بر رویِ دیتاهای بدونِ برچسب اعمال شده و سعی در اسخراج الگو و ساختِ مدل ‌می‌نماید. در واقع در این نوع یادگیری، هدف دسته‌بندیِ داده‌هاست و کشفِ ساختاری خاص در داده‌ها. به همین دلیل که کاربر نظارتی بر مدل ندارد و مدل خود به تنهای به کشفِ الگو بر روی داده‌های بدونِ برچسب می‌پردازد، به آن یادگیریِ بدونِ نظارت می‌گویند.

1. Supervised Learning

این روشِ یادگیری در واقع مربوط به یادگیریِ ماشینی است که یک سری داده‌های برچسب‌گذاری شده‌ی ورودی-خروجی را که به آنها نمونه‌های آموزشی می‌گویند را دریافت کرده و با استفاده از آنها تابعی را که وظیفه‌ی آن نگاشتِ ورودی به خروجی است، فرا می‌گیرد. به این دلیل به این نوع از یادگیری، نظارت شده می‌گویند که یک ناظر باید باشد که اطلاعات را در اختیارِ یادگیرنده قرار دهد و هدف از این نوع یادگیری، اکتشافِ ارتباطِ ورودی و خروجی است.

1. Semi-Supervised Learning

این روشِ یادگیری در واقع هم از داده‌های برچسب‌گذاری شده و هم از داده‌های بدون برچسب به جهتِ بالا بردنِ دقت در یادگیری استفاده می‌کند. در این روش تعداد از داده‌ها برچسب دارند و تعدادی بدونِ برچسب هستند و سعی می‌کنیم همه‌ی داده‌ها را به صورتِ برچسب گذاری شده دسته‌بندی کرده و خود را به supervised learning نزدیک‌تر نماییم.

1. Outlier

این نوع داده در واقع یک (یا یک سری) داده‌ی آماری است که از سایرِ داده‌های آماریِ متعلق به یک نمونه‌ی تصادفی، فاصله‌ی غیر عادی داشته باشد. در واقع این داده نشان‌دهنده‌ی غیرعادی بودنِ یکی از مشاهدات یا به عبارتی یکی از sample هاست (دقت شود که این داده خطا نیست و با نویز فرق دارد، این داده یک داده‌ی غیر عادی در مجموعه است).

1. Dimension

دیمانسیون یا بعد در داده‌کاوی (به عبارت بهتر در فیلدِ آمار) به معنای مجموعه اطلاعاتی است که از یک رویدادِ قابلِ اندازه‌گیری، ثبت و البته ذخیره می‌شود. در واقع با استفاده از دیمانسیون مدلی را طرح می‌کنند که هر دیمانسیون یک مجموعه اطلاعات را در اختیار قرار داده است و به جهت سریع‌تر شدنِ بازیابیِ اطلاعات از یک رویداد کاربرد دارد.

1. Training, Validating and Testing Data

در این عبارت دو نوع داده عنوان شده است. داده‌های آموزش یا training data به جهتِ ساختِ مدل مورد استفاده قرار می‌گیرند و در واقع فرایندِ یادگیری برای ساختِ مدل با اطلاعاتِ موجود در این داده‌ها انجام می‌پذیرد. اما داده‌های ارزیابی و تست یا validating and testing data به جهت بررسیِ کیفیتِ ساختار و کاراییِ مدلی است که ساخته شده و در واقع داده‌های تست به جهت بررسیِ performance مدل و ارزیابی و اعتبارسنجی از مدل موردِ استفاده قرار می‌گیرند.

1. Data Warehousing

Data warehousing در واقع یک فرایند جمع‌آوری و مدیریت از داده‌ها است که از منایع مختلف تهییه می‌شوند. این فرایند از این جهت کاربردی است که مجموعه‌ای از داده‌ها از منابع ناهمگون در کنارِ یکدیگر قرار گرفته و می‌تواند برای تحلیلِ داده‌ها بکار گرفته شود. در واقع DW یک انبار از داده‌ها از منابع گوناگون است که برای تحلیل و گزارش‌دهی از داده‌ها کاربرد دارد.

1. Missing Values

مقادیرِ از دست رفته یا داده‌های از دست رفته در واقع زمانی مطرح می‌شوند که هیچ مقدار داده برای متغیر در یک مشاهده ثبت نشده باشد. این missing value ها می‌تواند بدلیل اشکالِ سیستماتیک در جمع‌آوریِ درستِ داده‌ها صورت گیرد.

1. Independent Variable

متغیرهایی که در یک مشاهده بکارگرفته می‌شوند بدین منظور که اثرِ سایرِ متغیرها را ارزیابی کنند، متغیرِ مستقل گویند و به متغیرهایی که به ارزیابیِ آنها می‌پردازیم متغیرهای وابسته می‌گویند. در واقع در مباحثِ آماریِ داده‌کاوی، یک سری متغیرها را بکار می‌گیرند و به آنها مقادیرِ مختلف را اعمال می‌کنند تا اثرِ بقیه‌ی متغیرها را به آنها متغیر وابسته می‌گویند ارزیابی کنند.

**پاسخ سوال 2**

Dimensionality Reduction در واقع با حذف ویژگی‌هایی که اهمیتِ بالایی ندارند گویند.

تکنیک‌های Dimensionality Reduction :

* **Principal Components Analysis (PCA)**
* **Forward Feature Construction**
* **Backward Feature Elimination**
* **Random Forests/Ensemble Trees**
* **Missing Values Ratio**
* **Low Variance Filter**
* **High Correlation Filter**

سوال خواسته که یکی از متد‌های فوق را توضیح دهیم که ما دو تا را تصمیم داریم به طور مختصر بیان کنیم.

* **Low Variance Filter**

در این متد داده‌هایی که واریانسِ کمی دارند از مجموعه حذف شده و از ان طریق کاهشِ ابعاد صورت می‌گیرد. در واقع همانطور که می‌دانیم واریانس معیاری برای نشان دادنِ پراکندگیِ داده‌هاست و وقتی واریانس من باشد نتیجه می‌گیریم که داده‌ها در این ویژگیِ بخصوص مشابه یکدیگر هستند و نتیجتا این ستون در دیتاست یا هر کالکشنِ دیگر در پایگاهِ داده، اطلاعاتِ چندانی را در خود ندارد و لذا می‌توان این ستونِ ویژگی را حذف کرد. به عنوان مثال داده‌های تصویری را در نظر می‌گیریم که در آنها zero padding به تعداد مشخصی به کار برده شده است.( zero padding معمولا در شبکه‌های عصبیِ عمیق برای اضافه کردنِ سطر و ستون‌های صفر به ماتریسِ تصویر برای مرزبندی انجام می‌شود.) حال فرض کنیم ستون‌های داده‌ی ما میانگینِ ستونی از رنگ‌های یک عکس است(البته ماتریسِ عکس رنگی سه بعدی است و اینجا باید یک تصویرِ سیاه و سفید را در نظر بگیریم) حال مثلا در این ستون در ردیفِ اول میانگینِ مقادیرِ صفر و یک از ستونِ اولِ تصویرِ اول است ولی چون در این تصاویر zero padding بکاربرده شده است لذا مقادیرِ تمامیِ سطرهای این ستون صفر بوده و مطمئنا چون برای همه‌ی تصاویر این مقدار برابر و مساوی با صفر است و واریانس را صفر خواهد کرد و نتیجتا این اطلاعات نیز برای تحلیلِ داده اغلب با اهمیت نخواهند بود و می‌توان این ستون را به کلی حذف نمود.

* **Missing Values Ratio**

یا به عنوانِ یک متدِ دیگر می‌توان به این متد اشاره کرد که ستونِ داده‌ای که در آن missing value به وفور رخ داده است مسلما دارای اطلاعاتِ مفیدی نخواهد بود چون در خیلی از مشاهدات به هر دلیل داده‌ای دریافت نشده و ذخیره‌سازی انجام نشده است و لذا بدونِ داده هم که دیگر تحلیلی وجود ندارد. نتیجتا می‌توان با استفاده از این متد، این گونه ستون از داده‌ها را هم حذف نمود.

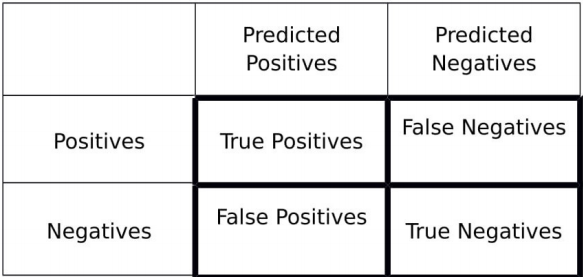
**Feature Selection VS Feature Extraction**

در واقع این دو مفهوم به کلی با یکدیگر تفاوت دارند. در استخراجِ ویژگی، یک سری ویژگی‌ها از داده تهییه شده و منجر به ایجادِ یک محصول جدید که همان فیچر آن مجموعه از داده‌ها است، می‌شود که نمونه‌ی بارزِ آن عملِ استخراجِ داده در شبکه‌های CNN است. اما در انتخابِ داده یا feature selection، داده‌های اضافی (داده‌هایی با ویژگی‌های کم اهمیت) از مجموعه‌ی داده‌ها، فیلتر شده و داده‌های با اهمیتِ بیشتر حفظ می‌شوند که میبینیم کاملا با استخراجِ ویژگی تفاوتِ زیادی دارد.

**پاسخ سوال 3**

این سوال از ما خواسته معیارهای ارزیابیِ دقت، فراخوانی و امتیازِ اِف را بر اساس ماتریسِ درهم ریختگی معرفی کنیم.

ماتریسِ درهم ریختگی به صورت زیر است:



ابتدا به تعریفِ هر یک از عناصرِ ماتریسِ در هم ریختگی می‌پردازیم:

* True Positives : داده هایی که به مجموعه تعلق داشته و تشخیص مدل در قبال این داده ها صحیح بوده است.
* False Positives : داده هایی که به مجموعه تعلق داشته ولی تشخیص مدل در قبال این داده ها اشتباه بوده است و آنها را خارج از مجموعه تشخیص داده.
* True Negatives : داده هایی که به مجموعه تعلق نداشته و تشخیص مدل در قبال این داده ها صحیح بوده و آن را خارج از مجموعه تشخیص داده است.
* False Negatives : داده هایی که به مجموعه تعلق نداشته ولی تشخیص مدل در قبال این داده ها اشتباه بوده است و آنها را متعلق به مجموعه تشخیص داده است.

حال سوال خواسته که معیارهای گفته شده را معرفی کنیم:

همانطور که مشاهده می‌کنید، recall نسبتی است بین تعداد اسنادِ مرتبطی که درست تشخیص داده شده‌اند بر تعداد کل اسنادِ مرتبط ولی precision در واقع نشان می‌دهد چه تعداد اسنادی که درست تشخیص داده شده‌اند مرتبط هستند و این نسبت را برآورد می‌کند. همچنین امتیازِ اف نیز با دخیل کردنِ recall و دقت، معیاری برای ارزیابیِ صحت را فراهم می‌کند.

**پاسخ سوال 4**

این که همبستگیِ دو متغیر صفر باشد به این معناست که هیچ ارتباط خطی بین دو متغیر وجود ندارد. در واقع هبستگی ارتباطِ خطی بین دو متغیر را اندازه‌گیری می‌کند.

اما باید توجه رد که این مفهوم با مفهومِ استقلال فرق می‌کند. در واقع دو متغیر که مستقل هستند هیچ شکلی از ارتباط بینِ آن‌ها برقرار نیست. در واقع فرض کنید که یک متغیر با توانِ دومِ دیگری به ازای نقاطی در ارتباط باشد، آنگاه همبستگیِ بینِ این دو متغیر صفر خواهد شد چون هیچ رابطه‌ی خطی بینِ این دو وجود ندارد ولی می‌دانیم که به واسطه‌ی رابطه‌ی غیر خطی بینِ آنها، این دو متغیر نمی‌توانند مستقل باشند. لذا نمی‌توانیم بگوییم که اگر دو متغیر همبستگیِ صفر دارند، لذا مستقل هستند. طبق توضیحاتی که دادیم، این جمله نادرست است.

**پاسخ سوال 5**

data cleaning که البته به آن data cleansing یا پاکسازی داده‌ها نیز می‌گویند، در واقع فرایندِ تشخیص و تصحیح یا حذف داده‌های خراب شده و ناصحیح را از جدولِ داده‌ها بر عهده دارد و در واقع به فرایندِ شناساییِ داده‌های ناقص، نادرست، ناصحیح یا قسمت‌های نادرستِ داده و سپس تصحیح یا حذفِ آنها اشاره می‌کند.

تجمیع داده یا data integration در واقع فرایند ترکیبِ داده از منابعِ مختلف و اراده‌ی آن به صورتِ یک نمای واحد به کاربر است. در واقع می‌توان گفت که برای داده‌های استخراج شده از منابع مختلف یک دیدگاه واحد وجود ندارد، لذا این داده‌ها پس از استخراج با الگوریتم‌های ترکیبِ داده که می‌توان از جمله‌ی آنها الگوریتم‌های owa و بیزین و کالمن فیلتر را معرفی کرد، داده‌ها را در کنارِ یکدیگر ترکیب کرده و یک دیدگاه که برآیندِ تجمیعِ داده هاست، به کاربر ارائه دهیم. به این تکنیک data integration می‌گویند.

در data transaction در واقع داده را از یک ساختار به یک ساختارِ دیگر تبدیل می‌کنیم. لذا یک تعریف کلی برای این تکنیک بدین صورت است که data transaction فرایندی است برای تغییرِ فرمت، ساختار یا مقدارِ داده به هر نحوی که می‌تواند یک نگاشت ساده باشد تا هر تبدیلِ دیگری. این تکنیک در مباحث تلفیق داده از اهمیت بالایی برخوردار است.

**پاسخ سوال 6**



مرحله اول

|  |  |
| --- | --- |
| Count | Item |
| 4 | نان |
| 2 | الویه |
| 2 | پنیر |
| 3 | کره |
| 4 | مربا |

همگی موارد فوق دارای support بیشتر از minsupport دارند. لذا هیچ یک در این مرحله حذف نمی‌شوند.

مرحله دوم

|  |  |
| --- | --- |
| Count | Item |
| 2 | (نان و الویه) |
| 1 | (نان و پنیر) |
| 2 | (نان و کره) |
| 2 | (نان و مربا) |
| 1 | (الویه و پنیر) |
| 0 | (الویه و کره) |
| 0 | (الویه و مربا) |
| 0 | (پنیر و کره) |
| 1 | (پنیر و مربا) |
| 3 | (کره و مربا) |

مرحله سوم

|  |  |
| --- | --- |
| Count | Item |
| 0 | (نان و الویه و کره) |
| 0 | (نان و الویه و مربا) |
| 2 | (نان و کره و مربا) |

در هر مرحله آیتم‌های پر تکرار را نگه داشته و بر روی غیر، خطِ قرمز کشیدیم. لذا به صورتِ جمع بندی آیتم‌های پرتکرار به صورت زیر است.

حال تمامی قواعد انجمی را می‌نویسیم. (دقت شود که آنهایی که حد آستانه را پاس نکرده بودند دیگر ذکر نمی‌شوند)

توجه شود که شرط minsupport را قبلا بررسی نموده بودیم.

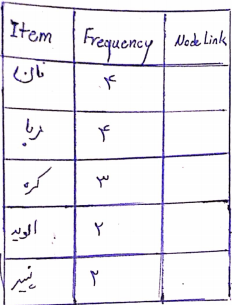
حال طبق خواسته سوال، تمامی موارد را به ترتیب اطمینانِ آنها می‌چینیم.

**پاسخ سوال 7**

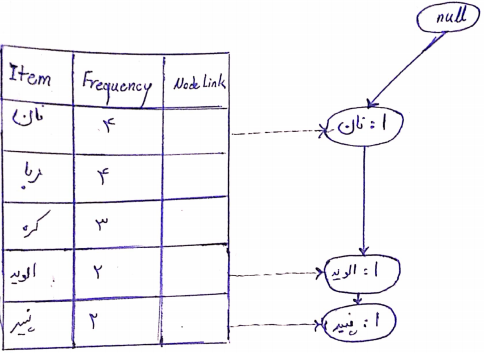
برای ساخت درخت fp-tree باید ابتدا آیتم‌های پر تکرار را بیابیم و تکرارِ هر یک را محاسبه نماییم که در قسمت 6، این کار را انجام دادیم.

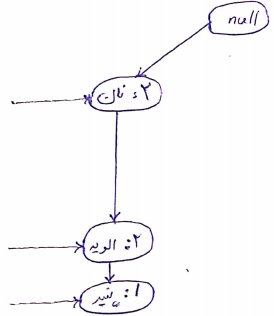
**بخش الف**

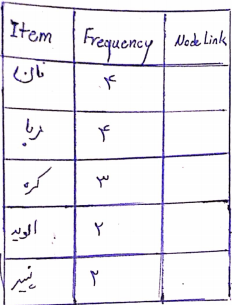
حال باید آیتم ها را به ترتیبِ تعداد تکرار از بزرگ به کوچک مرتب نماییم. لذا بدین صورت در می‌آید.

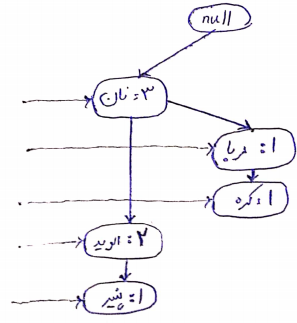


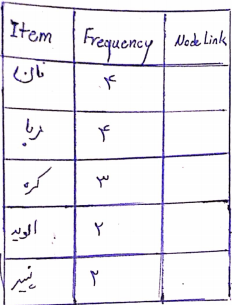
حال درخت را با first transaction آغاز به ساخت می‌کنیم.

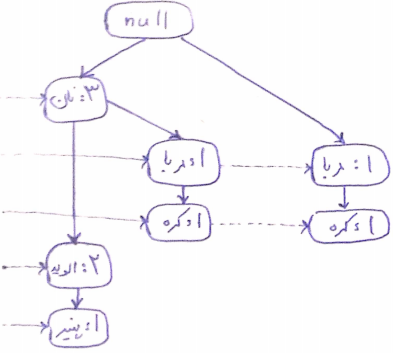
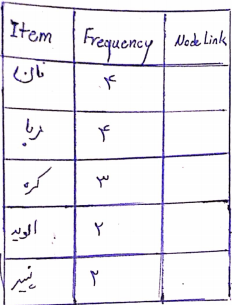


اکنون second transaction را اعمال می‌نماییم. (تفاوت در اندیس‌های مقابل هر item نسبت به درختِ قبلی ایجاد شده است)

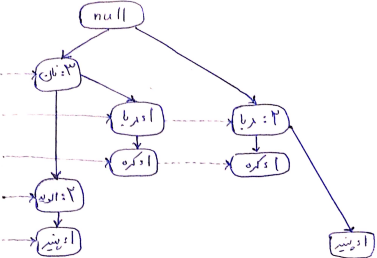
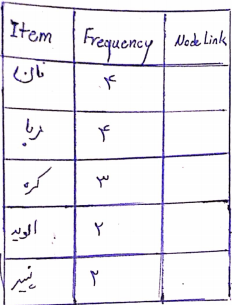


اکنون third transaction را اعمال می‌نماییم.

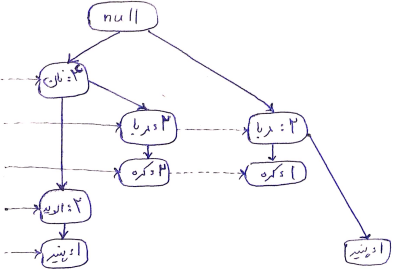
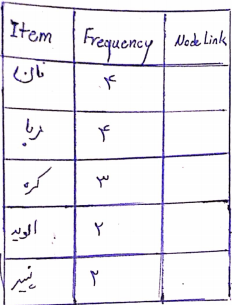


اکنون 4th transaction را اعمال می‌کنیم.

کنون 5th transaction را اعمال می‌کنیم.



و آخرین transaction اعمال شده به درخت به صورت زیر خواهد بود و درخت در این مرحله تکمیل می‌شود.





**بخش ب**

اکنون که fp-tree را ساختیم می‌توانیم به مانینگِ پترن‌های پرتکرار بپردازیم.

پس ما تا این لحظه قسمتِ اول این الگوریتم را که ساخت fp-tree بود انجام دادیم. حال به مانینگ می‌پردازیم.

مرحله اول:

|  |  |
| --- | --- |
| Conditional pattern base | Items |
| {{ : 1الویه , نان} , { : 1مربا}} | پنیر |
| { : 2نان} | الویه |
| {{ : 2مربا , نان} , { : 1مربا}} | کره |
| { : 2نان} | مربا |
| - | نان |

مرحله دوم: ( آنهایی که مین ساپورت را پاس نمی‌کنند یادداشت نمی‌شوند)

|  |  |  |
| --- | --- | --- |
| Conditional frequent pattern tree | Conditional pattern base | Items |
| - | {{ : 1الویه , نان} , { : 1مربا}} | پنیر |
| { : 2نان} | { : 2نان} | الویه |
| {{ : 3مربا},{ : 2نان}} | {{ : 2مربا , نان} , { : 1مربا}} | کره |
| { : 2نان} | { : 2نان} | مربا |
| - | - | نان |

مرحله سوم

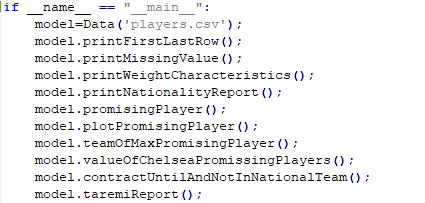
|  |  |
| --- | --- |
| Frequent pattern generated | Items |
| - | پنیر |
| {« : 2الویه , نان»} | الویه |
| {«: 2کره , مربا , نان» ,« : 2کره , نان» , « :3کره , مربا»} | کره |
| {« : 2مربا , نان»} | مربا |
| - | نان |

تمامی مجموعه‌های پرتکرارِ بیش از یکی توسطِ الگوریتمِ فوق پیدا شد و از طرفی مجموعه‌های تک آیتمه‌ی پرتکرار را نیز در ابتدای الگوریتم در ساخت fp-tree بدست اوردیم لذا مجموعه آیتم های پر تکرار ما به صورت زیر است.

|  |  |
| --- | --- |
| Count | Item set |
| 4 | نان |
| 4 | مربا |
| 3 | کره |
| 2 | الویه |
| 2 | پنیر |
| 3 | کره , مربا |
| 2 | کره , نان |
| 2 | مربا , نان |
| 2 | الویه , نان |
| 2 | کره , مربا , نان |

**پاسخ بخش پیاده‌سازی**

برای حل سوالات این بخش یک کلاس با نامِ Data ایجاد کردیم و در این کلاس ده تابع تعریف نمودیم که هر تابع، پاسخِ یک بخش از سوال است. لذا مدلی از کلاس درست کرده و تابع‌های کلاس را فراخوانی نمودیم.



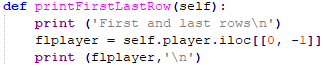
حال به پاسخ سوالات می‌پردازیم:

**بخش 1**

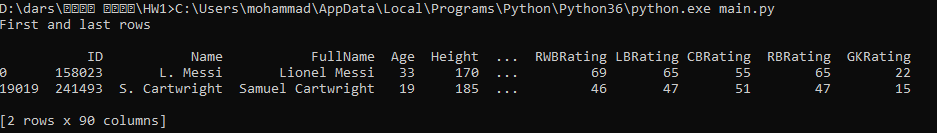
همانطور که در کد قبل دیدیم، فایلِ اکسل به ورودی کلاس داده می‌شود.



حال توسط تابع printFirstLastRow موارد خواسته شده یعنی سطر اول و آخر را نمایش می‌دهیم.

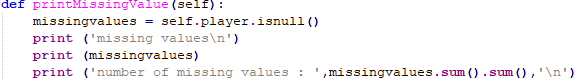


پس از دستور iloc استفاده کرده و خروجی به صورت زیر در ترمینالِ ویندوز، پرینت می‌شود.

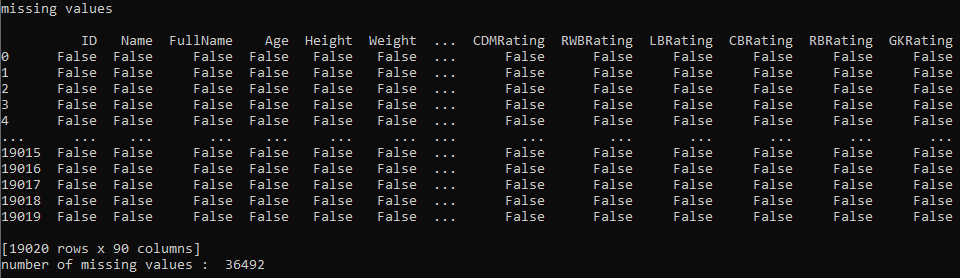


**بخش 2**

برای مشخص شدنِ missing value ها از دستورِ isnull() استفاده کردیم. با این دستور هر جایی که مقداری miss شده باشد، به کاربر true نمایش داده و هر جایی از جدول که دیتا موجود باشد، کاربر false دریافت می‌نماید.



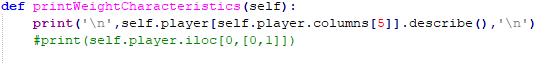
نتیجتا خروجی به صورت زیر خواهد شد. تعداد missing values را نیز به کاربر نمایش دادیم.



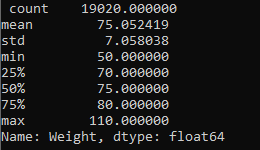
پس تعداد missing value ها برابر با 36492 داده است.

**بخش 3**

برای حل این بخش همانطور که از ما خواسته شده بود از امکاناتِ لایبریِ پانداس در دیتا ماینیگ استفاده کنیم. لذا ما نیز از دستورِ describe استفاده کرده و تمامی مشخصات آماری مربوط به وزنِ بازیگران را استخراج نمودیم.



نهایتا خروجی به صورت زیر خواهد شد.



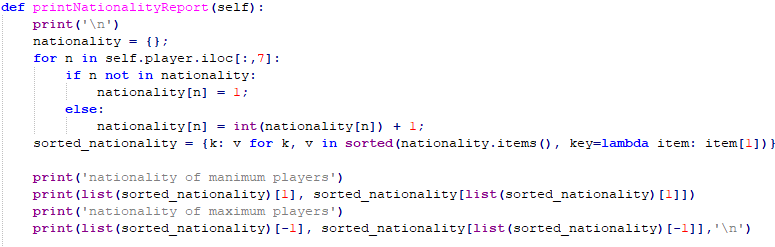
آنها را در جدول زیر بازنویسی می‌کنیم.

|  |  |
| --- | --- |
| **75.052419** | Mean Weight |
| **50** | Min Weight |
| **110** | Max Weight |

**بخش 4**

در این بخش، مسأله از ما خواسته است که کشوری را که دارای بیشتری بازیکنان و کمترین بازیکنان است را بیابیم. لذا به سادکی یک دیکشنری با نامِ ملیت تعریف نمودیم و کلیدِ آن را ملیت و مقدار آن را از صفر شروع کرده و به ازایِ هر بار خواندنِ آن، یکی به آن ملیت اضافه نمودیم. ( البته بهتر بود اسمش را کشور قرار می‌دادم ☺)

سپس دیکشنری را بر حسبِ مقادیرِ value ها سورت کرده و ابتدا و انتهای آن را پرینت کردیم.



نهایتا خروجی به صورتِ زیر قابل روئت است.

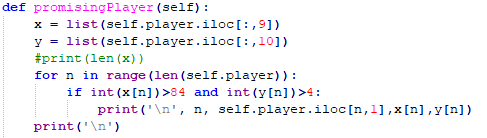


آن را مجددا در جدول زیر باز نویسی می‌کنیم.

|  |  |
| --- | --- |
| Chad : 1 | nation of minimum players |
| England : 1706 | nationality of maximum players |

**بخش 5**

در این بخش مسأله از ما خواسته است که آینده‌دارترین بازیکنان را بر اساسِ معیارِ potential بالای 84 و growth بالای 4 را از داده‌ها استخراج کنیم. و آن را گزارش کنیم.



در اینجا چون نیاز به مفایسه تک تک داده‌های ستون بود، آنها را به صورت list قرار داده و مقایسه را انجام دادیم.

نتایج این بازیکان به صورت زیر است (دو عدد سمت راست بازیگر، عدد اول potential و عدد دوم growth است.)

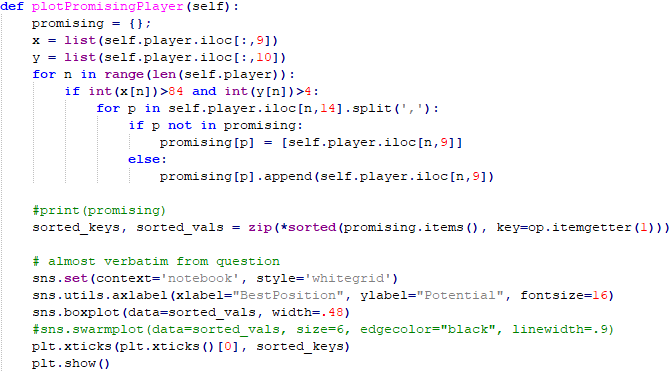
عدد شمت چپ ترین نیز شماره در جدول است.

|  |  |  |
| --- | --- | --- |
| 1873 M. Boadu 86 11  1970 Arthur Cabral 85 10  2082 Nuno Mendes 86 12  2100 E. Barco 85 11  2128 Gabriel Martinelli 85 11  2218 N. Bustos 85 11  2226 R. Sessegnon 85 11  2327 C. Hudson-Odoi 86 12  2354 Gustavo Assunção 86 12  2398 G. Plata 86 12  2406 W. Saliba 87 13  2416 M. Solomon 86 12  2439 G. Diangana 85 11  2451 G. Reyna 88 14  2467 A. Hložek 87 13  2538 Brahim 86 12  2609 R. Aït Nouri 86 13  2612 J. Bellingham 88 15  2634 Romário Baró 85 12  2639 A. Urzi 88 15  2659 Pedri 89 16  2705 Eric García 85 12  2769 Barrenetxea 85 12  2823 C. De Ketelaere 86 13  2903 Marcos Antonio 86 13  2910 W. Fofana 85 12  2911 F. Wirtz 88 15  2923 Rafael Camacho 86 13  3010 Evanilson 86 13  3115 Tomás Tavares 85 12  3240 L. Pellegrini 86 14  3341 G. Tsitaishvili 86 14  3412 Nuno Tavares 85 13  3434 M. Longstaff 85 13  3614 Y. Verschaeren 85 13  3690 Daniel Bragança 86 14  3691 Eduardo Quaresma 85 13  3709 N. Madueke 86 14  4001 F. Pellistri 87 16  4058 Reinier 86 15  4080 V. Korniienko 85 14  4266 B. Gilmour 86 15  4336 T. Nianzou 85 14  4548 P. De la Vega 85 14  4583 J. Willock 85 14  4617 H. Traorè 86 15  4618 J. Doku 88 17  5379 J. Gvardiol 86 16  5522 K. Adeyemi 86 17  5702 A. Velasco 85 16  5876 R. Cherki 88 19  5908 C. Tzolis 87 18  6061 E. Smith Rowe 85 16  6064 Abel Ruiz 86 17  6069 Sergio Gómez 85 16  6200 Joelson Fernandes 87 18  6425 Fábio Silva 85 16  6609 C. Jones 86 18  6803 J. Musiala 85 17  6834 N. Williams 85 17  6904 M. Vandevoordt 86 18  7918 H. Elliott 85 18  8568 L. Stergiou 86 19  8750 T. Parrott 85 19  8970 S. Esposito 86 20  10060 X. Simons 85 20  10650 Y. Demir 86 21  12472 J. Branthwaite 85 22  12798 L. Netz 85 22  14022 T. Nakai 86 24  14290 B. Arrey-Mbi 85 24 | 583 Dani Olmo 87 8  589 E. Tapsoba 88 9  590 Antony 88 9  622 C. Pavón 85 6  623 S. Berge 85 6  634 R. Bentancur 85 6  639 D. Ćaleta-Car 85 6  735 Trincão 88 10  739 Emerson 88 10  740 D. Kulusevski 88 10  753 L. Díaz 87 9  755 M. Senesi 86 8  756 D. Malen 86 8  760 I. Sarr 87 9  765 C. Stengs 86 8  766 Pedro Gonçalves 87 9  769 I. Konaté 88 10  770 R. James 86 8  773 N. De la Cruz 85 7  778 G. Montiel 85 7  799 E. Camavinga 89 11  824 T. Abraham 85 7  825 D. McNeil 85 7  839 A. Meret 86 8  843 K. Tierney 86 8  895 Bruno Guimarães 86 8  905 K. Diatta 85 8  909 Jovane Cabral 86 9  927 D. Szoboszlai 87 10  932 M. Greenwood 89 12  943 M. Guendouzi 86 9  968 S. Tonali 89 12  992 E. Palacios 85 8  994 Jovane Cabral 86 9  995 O. Wijndal 85 8  997 S. Tonali 89 12  998 M. Olivera 85 8  999 Galeno 85 8  1000 K. Diatta 85 8  1002 D. Szoboszlai 87 10  1003 M. Guendouzi 86 9  1005 Luiz Felipe 85 8  1011 J. Bijlow 86 9  1012 J. David 88 11  1013 U. Račić 85 8  1036 M. Kudus 87 10  1053 M. Greenwood 89 12  1073 M. Ihattaren 86 9  1074 S. Chukwueze 86 9  1084 Galeno 85 8  1096 M. Olivera 85 8  1110 E. Palacios 85 8  1116 J. David 88 11  1124 J. Boga 86 9  1127 M. Kudus 87 10  1138 S. Chukwueze 86 9  1145 M. Ihattaren 86 9  1151 J. Veerman 85 8  1153 J. Veerman 85 8  1161 M. Caqueret 86 10  1194 M. Demiral 85 9  1229 L. Martínez Quarta 85 9  1235 Tete 86 10  1279 B. Soumaré 85 9  1315 B. Saka 88 12  1357 W. McKennie 85 9  1385 Diogo Dalot 85 9  1386 Ansu Fati 90 14  1422 F. Tomori 85 9  1435 B. White 87 11  1438 N. Zaniolo 86 10  1459 Pedro Neto 85 9  1464 O. Kabak 86 10  1520 I. Sangaré 85 10  1541 A. Mac Allister 85 10  1548 J. Todibo 86 11  1566 Óscar 86 11  1583 J. Reine-Adélaïde 85 10  1584 S. Dest 86 11  1613 T. Almada 89 14  1628 T. Kubo 88 13  1666 Riqui Puig 88 13  1726 Pedrinho 86 11  1728 A. Lunin 87 12  1732 N. Pérez 85 10  1866 R. Gravenberch 89 14 | 12 K. Mbappé 95 5  45 T. Alexander-Arnold 92 5  46 J. Sancho 93 6  68 L. Sané 90 5  70 F. de Jong 90 5  71 G. Donnarumma 92 7  72 M. Rashford 91 6  75 M. de Ligt 92 7  76 K. Havertz 93 8  77 E. Haaland 92 7  115 Oyarzabal 89 5  118 L. Martínez 91 7  141 N. Süle 88 5  149 M. Ødegaard 89 6  150 F. Valverde 90 7  154 F. Mendy 88 5  156 A. Wan-Bissaka 88 5  159 O. Dembélé 89 6  160 A. Hakimi 88 5  161 J. Gomez 88 5  162 T. Strakosha 88 5  188 Gayà 88 5  257 Fabián 88 6  259 Reguilón 87 5  260 João Félix 93 11  261 H. Aouar 89 7  264 G. Lo Celso 87 5  265 S. Bergwijn 87 5  266 P. Kimpembe 87 5  268 Rúben Neves 87 5  274 Richarlison 88 7  276 A. Davies 89 8  278 T. Souček 86 5  279 Cucurella 89 8  280 Rúben Dias 87 6  281 Renan Lodi 87 6  282 D. Livaković 86 5  283 M. Diaby 88 7  323 O. Vlachodimos 86 5  353 D. Calvert-Lewin 87 6  354 Everton 86 5  357 N. Barella 88 7  362 C. Pulisic 87 6  364 Gonçalo Guedes 87 6  379 V. Tsygankov 86 6  380 M. Gómez 85 5  381 Éder Militão 87 7  382 N. Vlašić 86 6  383 Ferran Torres 88 8  384 Pau Torres 86 6  385 J. Koundé 86 6  421 David Neres 85 5  432 Vinícius Jr. 93 13  433 M. Arambarri 85 5  434 T. Ndombele 87 7  435 Palhinha 85 5  437 L. Bailey 85 5  451 D. Upamecano 90 10  453 F. Neuhaus 86 6  454 M. Thuram 86 6  458 A. Saint-Maximin 86 6  462 L. Klostermann 85 5  465 Diogo Jota 86 6  466 N. Mukiele 87 7  479 Unai Simón 86 6  480 C. Söyüncü 85 5  481 C. Nkunku 86 6  482 L. Jović 85 5  483 T. Hernández 85 5  484 M. Mount 87 7  486 D. Henderson 87 7  488 H. Barnes 85 5  516 Renato Sanches 85 6  518 V. Osimhen 87 8  519 Gabriel 85 6  520 A. Isak 86 7  523 Carlos Soler 85 6  524 D. Rice 86 7  525 J. Ikoné 86 7  526 B. Kamara 85 6  529 P. Foden 88 9  532 D. Zagadou 86 7  551 M. Locatelli 85 6  553 P. Estupiñán 86 7  558 Adama Traoré 85 6  578 Matheus Cunha 87 8  580 Junior Firpo 85 6  582 Rodrygo 90 11 |

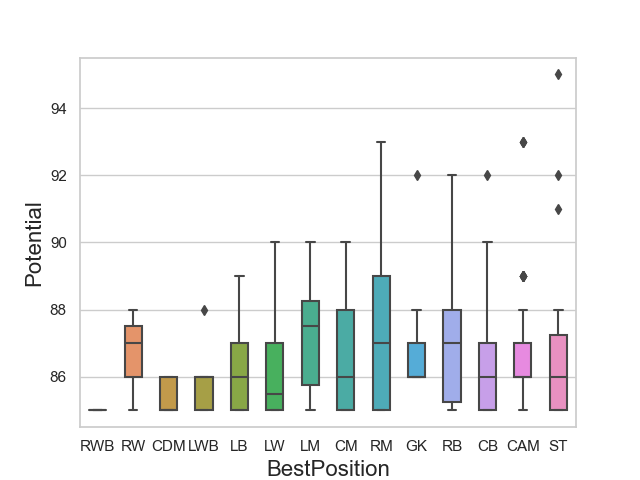
**بخش 6**

در این قسمت خواسته شده که نمودارِ بازیکنانِ آینده دار را بر اساس موقعیتشان گزارش کنیم. لذا بدلیل حجم بالا از ابزار boxplot استفاده کردیم و روند حل هم بدین صورت است که یک دیکشنری خالی را تعریف کردیم و موقعیت به صورت key و potential ها هم به صورت value و در قالبِ یک لیست به آن اضافه نمودیم و تا انتهای پیمایشِ ستونِ مربوط به position ها، این روند را ادامه دادیم.

کد به صورت زیر است.( بعد از آن شکل رسم شده قرار گرفته است.)

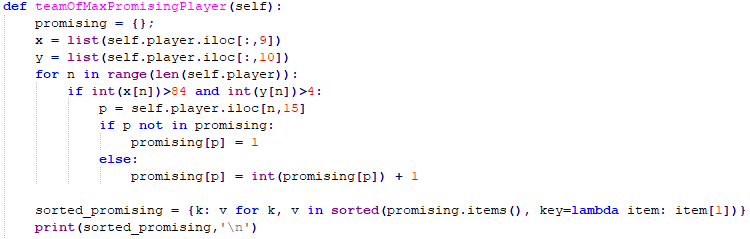


نهایتا خروجی به صورت زیر plot نمودیم.



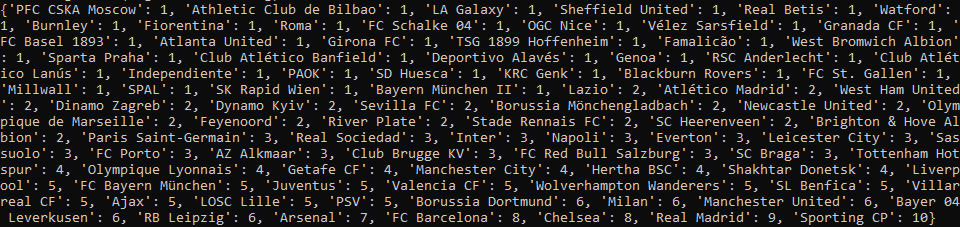
**بخش 7**

در این بخش از ما خواسته شده است که بررسی کنیم کدام باشگاه دارای بیشترین تعداد بازیکنانِ آینده دار است. لذا در اینجا نیز یک دیکشنری تعریف نموده و key آن را برابر با باشگاه ها یا همان club قرار داده و value آن را به ازای هر بار پیمایشِ مربوط به آن باشگاه در پیشمایشِ بازیکنانِ آینده دار، یکی اضافه نموده و در انتها که پیمایش به صورتِ کامل انجام شود، نهایتا تعداد بازیکنانِ آینده‌دارِ متعلق به هر باشگاه بدست آمد.



همچنین در آخر نیز عملیاتِ sort را روی value های این دیکشنری انجام دادیم.

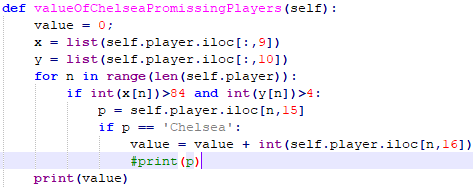
نهایتا خروجی به صورت زیر است:



دیکشنریِ فوق به صورتِ سورت شده پرینت شده است و لذا همانطور که مشاهده می‌فرمایید، بیشترین تعداد بازیکنانِ آینده دار با تعداد 10 بازیکن، متعلق به Sporting CP است.

**بخش 8**

در این بخش مجموعِ ارزشِ بازیکنانِ آینده دارِ چلسی را از ما خواسته است. لذا بحث استخراج بازیکنان آینده دار در این تابع نیز مثل روال گذشته انجام می‌شود ولی در هر بار یک مقایسه صورت می‌گیرد که آیا باشگاهِ مربوط به این بازیکن چلسی است یا خیر. در صورتی که چلسی باشد مقادیر با یکدیگر جمع کرده و در نهایت به کاربر نمایش داده می‌شود. این متغیر که مجموعِ ارزش‌ها را در آن می ریزیم را با value نامیده ایم.



همانطور که مشاهده می‌نمایید، داده‌ها از ستونِ 16 یعنی ValueEUR خوانده شده‌اند.

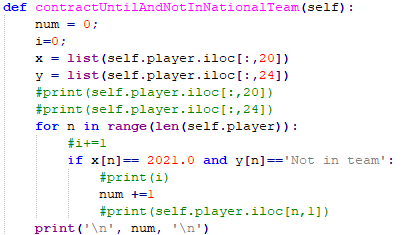
نهایتا مقدار value به صورت زیر در می‌آید.



پس مجموع ارزشِ بازیکنانِ چلسی برابر با 293900000 خواهد بود.

**بخش 9**

این قسمت دقیقا مانندِ قسمتِ 5 است با این تفاوت که ستون‌ها NationalTeam و contractUtil می‌باشد و البته مقایسه در اینجا مقایسه‌ی بینِ string هاست مگر مراحل مانند قسمت 5 است و در اینجا نتایج را نمایش می‌دهیم.



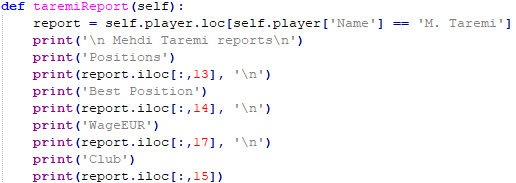
در نهایت تعداد این بازیکنان را در متغیرِ num ریخته و آن را پرینت می‌کنیم.



لذا تعداد بازیکنانی که در سال 2021 قراردادشان با باشگاهشان تمام میشود و همچنین در تیم ملی کشورشان حضور ندارند برابر با 6727 بازیگر است.

**بخش 10**

این بخش که بخشِ آخرِ مسأله بود از ما خواسته که موقعیت، درامد و باشگاه فعلی مهدی طارمی را پرینت کنیم. البته ما از بهترین موقعیتِ آن نیز پرینت گرفتیم. در این قسمت ابتدا سطر مربوطه به مهدی طارمی را در قسمت نام‌ها Name جستجو کردیم (M. Taremi) و نهایتا ستون‌های خواسته شده در صورت سوال را استخراج کرده و به کاربر نمایش دادیم.



نهایتا خروجی به صورت زیر بدست آمد.



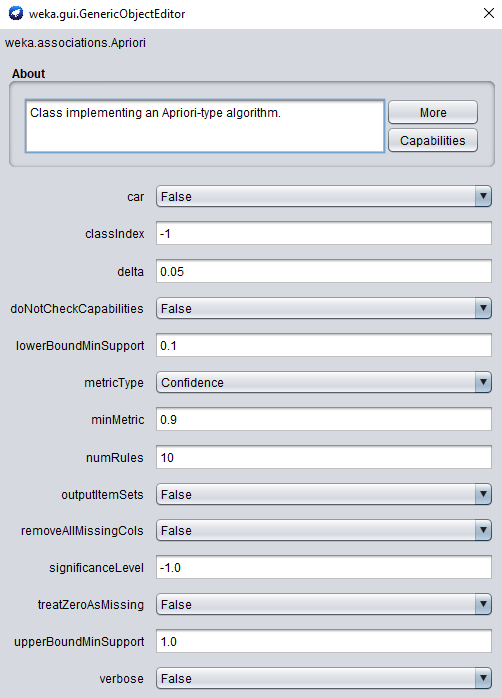
چون دو سطر جدول مربوط به مهدی طارمی بود لذا هر کدام دوتایی چاپ شده است.

نتایج را مجددا در زیر بازنویسی می‌کنیم.

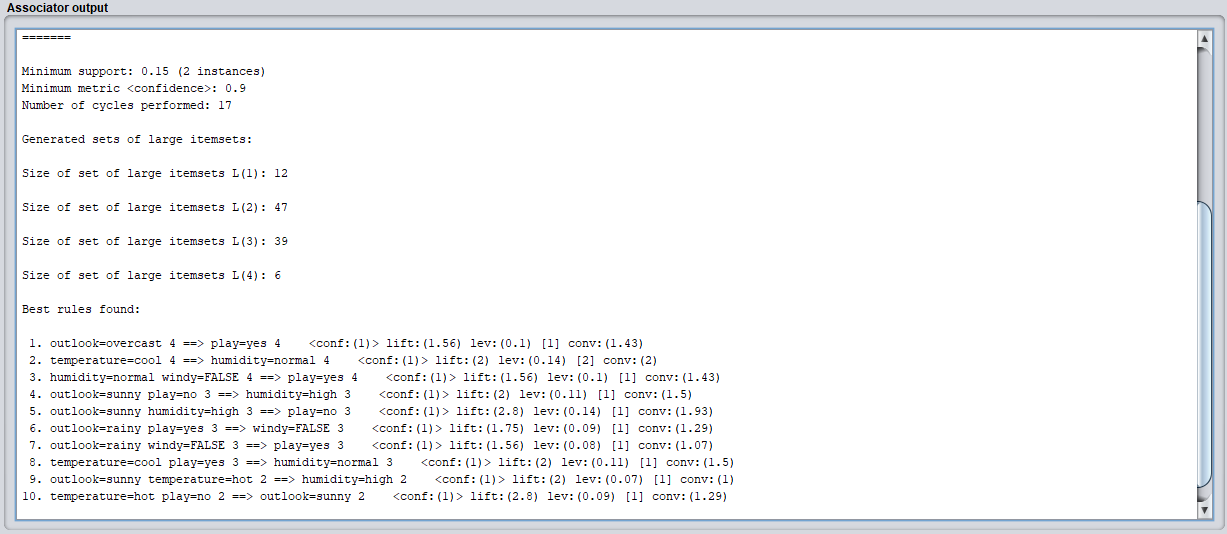
|  |  |
| --- | --- |
| ST,CF | Position |
| ST | Best Position |
| 16000 | WageEUR |
| FC Porto | Club |

**پاسخ قسمت قوانین انجمنی**

ابتدا نرم افزار وکا را نصب کرده و حالت دیفالتِ آن را در نظر می‌گیریم.



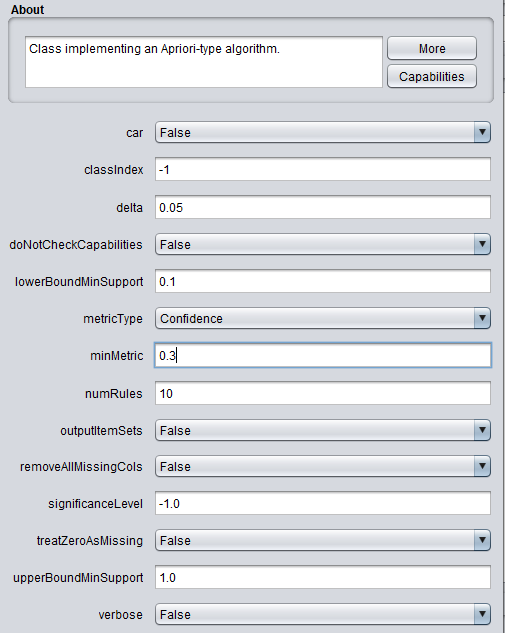
در حالتِ دیفالتِ نرم‌افزار نتایج به صورت زیر است.



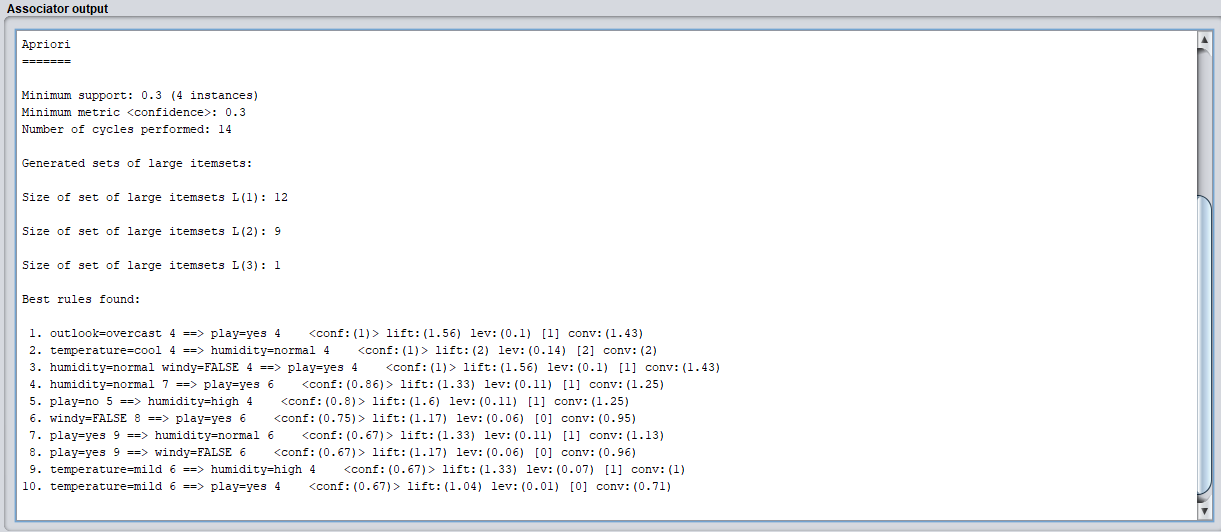
حال 5 پارامترها تغییر داده و تأثیرِ آنها در خروجی را بررسی می‌کنیم.

**حالت اول (تغییر minconf)**

پارامترِ minconf را از 0.9 به 0.3 کاهش داده و نتیجه را بررسی می‌کنیم.



خروجی به صورت زیر است.



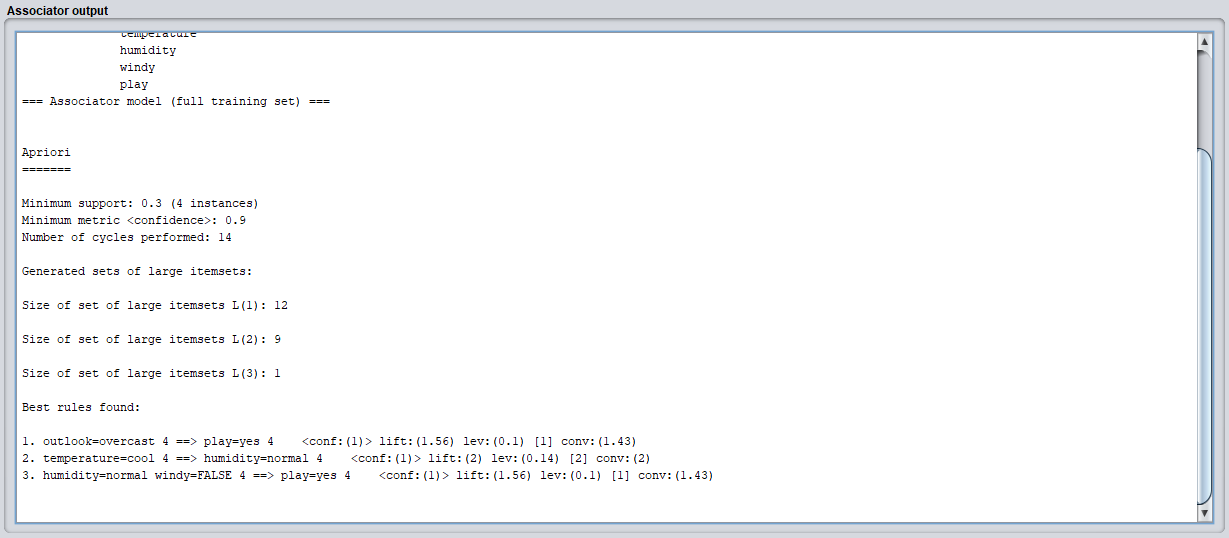
ابتدا مشاهده می‌کنیم که با کاهشِ minconf، تعداد سیکل‌های پردازش شده از 17 تا به 14 عدد کاهش یافته است. همچنین اگرچه در سایزِ آیتم‌سِت L(1) تغییری حاصل نشده است ولی سایز آیتم‌ست‌های بعدی، کاهشِ چشمگیری داشته است.

به گونه‌ای که L(2) و L(3) و L(4) ، هر یک به ترتیب از 47 و 39 و 6 به 9، 1 و صفر کاهشِ سایز داشته است که البته تأثیرِ آن را می‌توان به اثری که بر روی افزایشِ minsup گذاشته است اشاره کرد و این مورد کاملا قابلِ پیش‌بینی بود.

یعنی یکی از اثراتی که در این حالت نسبت به دیفالت مشاهده می‌شود، همین افزایشِ minsup از 0.15 به 0.3 است.

**حالت دوم (تغییر numRules)**

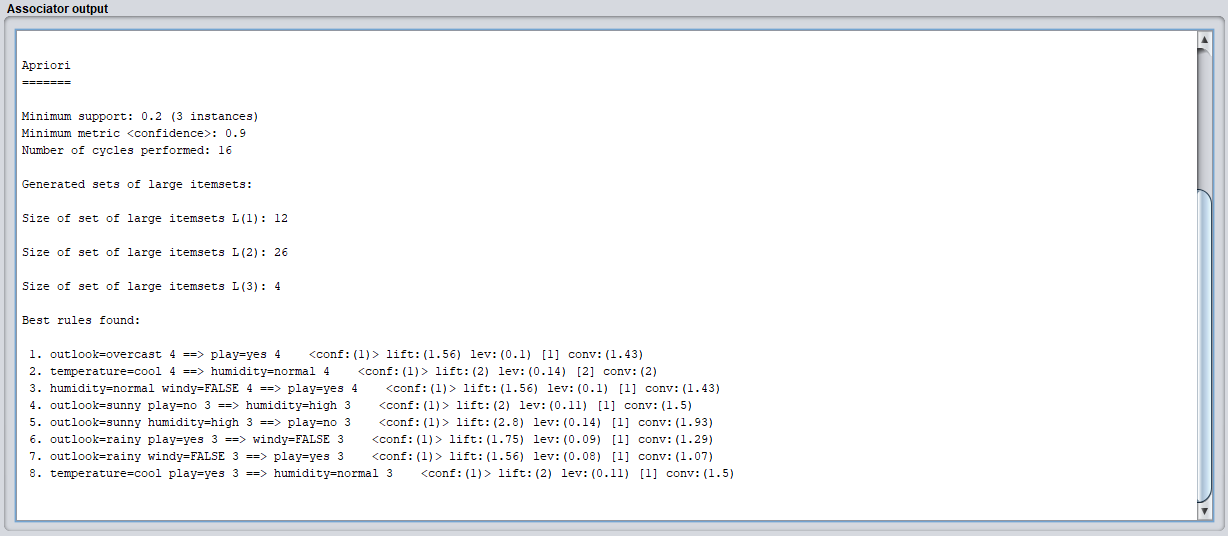
در این حالت تعداد قوانینِ انجمنی را از 10 عدد به 3 عدد کاهش می‌دهیم و نتیجه را می‌بینیم.



دقت بفرمایید که این حالت فقط در numRules با حالتِ دیفالت فرق دارد. لذا همانطور که مشاهده می‌کنیم مثل حالتِ اول تعداد سیکل های پردازش شده از 17 تا به 14 تا کاهش یافته و همچنین L(2) و L(3) و L(4) ، هر یک به ترتیب از 47 و 39 و 6 به 9، 1 و صفر کاهشِ سایز داشته است که البته تأثیرِ آن را می‌توان باز به minsup ارجاع داد که از 0.15 به 0.3 افزایش یافته است.

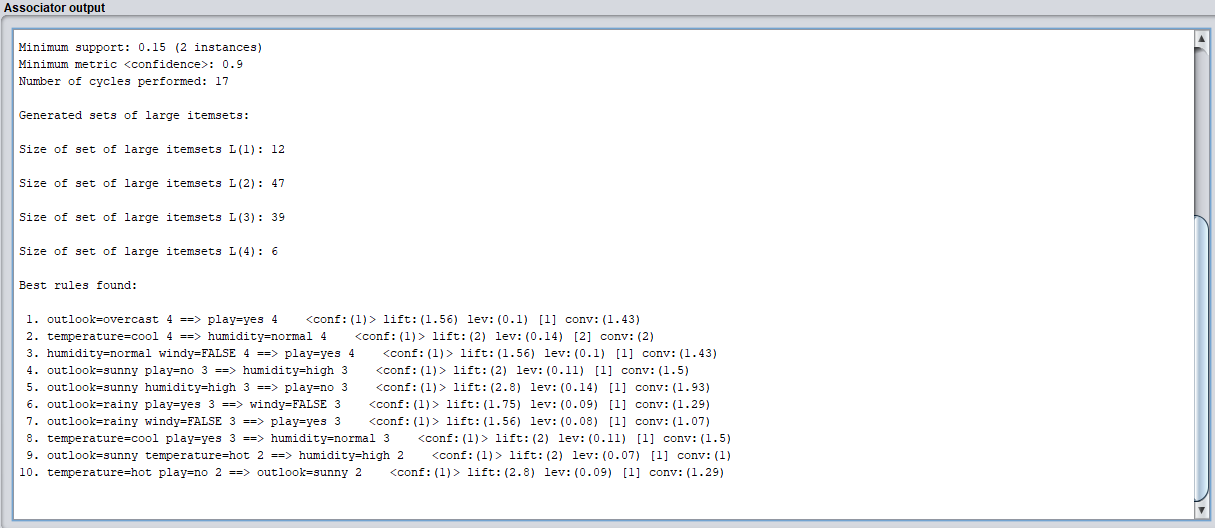
**حالت سوم (تغییر lowerBoundMinSupport)**

در این قسمت نیز تنها تفاوتی که با حالتِ دیفالت ایجاد نمودیم، افزایشِ حدِ پایینِ minsupport است که انتظار داریم در صورتی که باعثِ محدود کردنِ support سیستم شود که می‌شود (چون در حالتِ دیفالت minsupport را سیستم 0.15 می‌گذاشت)، لذا سایز مجموعه آیتم‌ست ها کاهش پیدا کند. حال نتیجه را می‌بینیم که آیا اینطور است یا خیر.



همانطور که مشاهده می‌فرمایید دقیقا پیش‌بینی ما درست بود. دقت کنید که بیشتر نباید این حد پایین را افزایش داد چون سایز تمامی مجموعه آیتم ست‌ها را کاهش داده تا به صفر برسند. حال به اثرات دیگر توجه کنید که سیکل‌های پردازش‌شده نیز در این حالت یکی کاهش یافته. همچنین تفاوتِ این حالت نسبت به دیفالت این است که اگرچه ما تعداد قواعد را 10 قرار داده بودیم ولی قواعدی که سیستم استخراج کرده برابر با 8 عدد است. این یعنی حد پایینِ minsupport در صورت افزایش احتمالا در کاهشِ پیدا کردنِ قواعد اثر می‌گذارد.

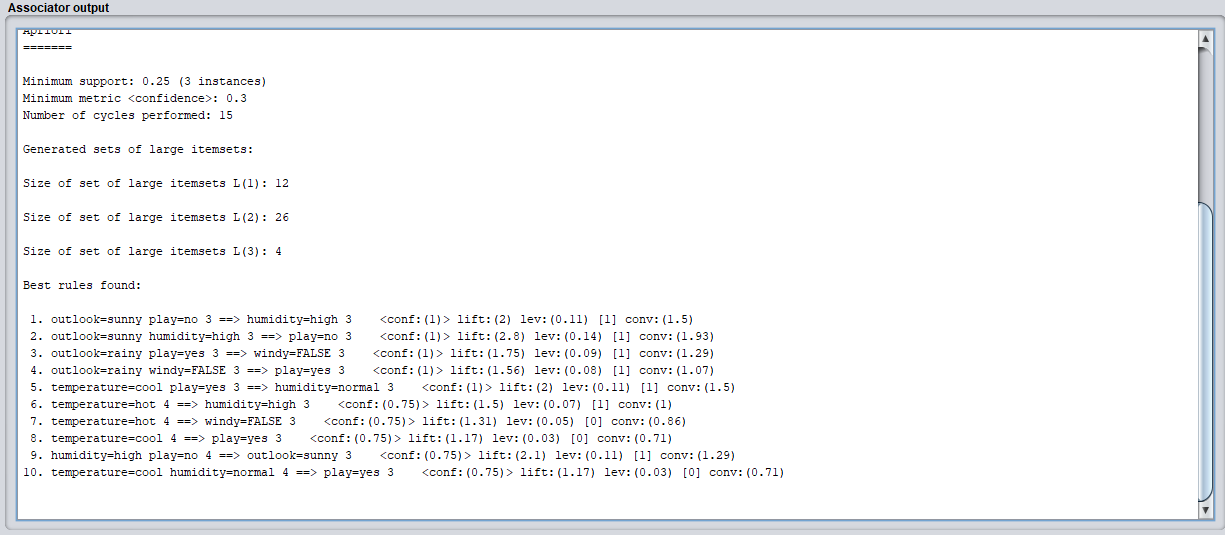
**حالت چهارم (تغییر removeAllMissingCols)**



همانطور که مشاهده می‌فرمایید این پارامتر تغییری در سیستم ایجاد نکره است. نه سایز آیتم ست‌ها نه قوانین و نه در تعیینِ پارامترِ minsup. لذا دو مورد در این سیستم از این نتیجه قابلِ برداشت است. اول اینکه سیستم missing value نداشته باشد. حالت دوم این می‌تواند باشد که این پارمتر تاثیری نمی‌گذارد. به نظر می‌رسد که منظور کلِ ستون با بقیه‌ی داده‌های موجود باشد لذا حالتِ یک محتمل‌تر و منطقی تر است. ولی به هر حال چیزی که در این پایگاه داده مطرح است این است که حذف missingValuesCols تاثیری بر نتایجِ ما در مقایسه با حالتِ دیفالت نگذاشته است.

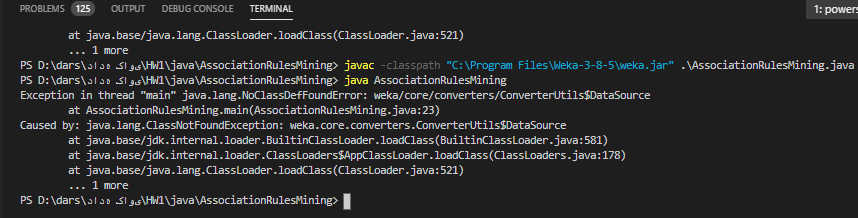
**حالت پنجم (تغییر upperBoundMinSupport)**

این تغییر حالت در پارامتر فوق و نسبت به حالت یک (نه حالتِ دیفالت) انجام گرفته است. این پارامتر را بر روی 0.2 قرار داده‌ایم و نتیجه به صورت زیر است.



همانطور که مشاهده می‌کنید تعداد سیکل‌های پردازش یافته از 14 به 15 افزایش یافته است و همچنین minsuppoer نیز به مقدار 0.05 کاهش داشته است. این طبیعتا تاثیرِ خود را در سایزِ آیتم ست ها می‌گذارد که همینطور هم هست. یعنی سایز L(2) و L(3) نسبت به حالتِ اول افزایش داشته است.

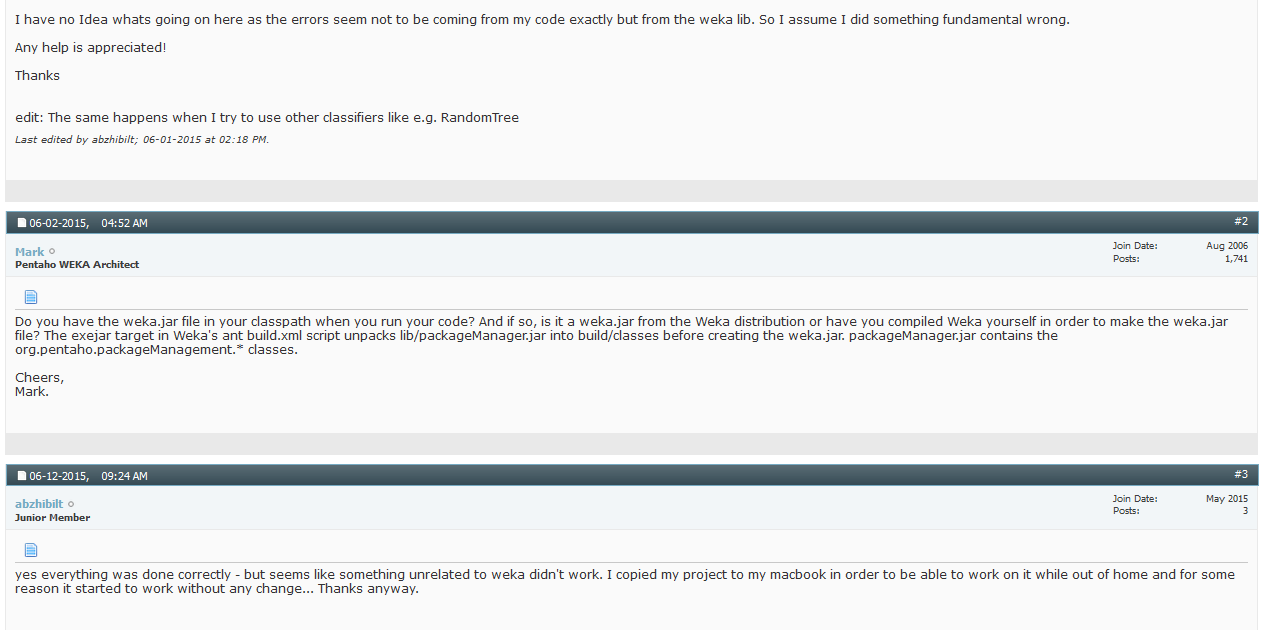
برای قسمت بعدی که آخرین قسمت سوالات تمرین است و مربوط به تست گرفتن با کد جاوا بود، اینجانب تمامی فایل‌ها به درستی ران گرفته و فایل class آنها را تولید کردم ولی مشکلی که ایجاد شده بود مربوط به فایل weka.rar بود و با این ارور مواجه می‌شدم. (مشاهده کنید)



همانطور که مشاهده می‌کنیم به ConverterUtils$DataSource گیر داده و عملیات ران را به طور انجام انجام نمی‌دهد.

لذا پس از سرچ فراوان و بررسی فایل‌های jar مختلفِ موجود از weka، سعی نمی‌دونم ارور را تصحیح کنم که بعد از یک روز به رفرنس پایین خوردم.

https://forums.pentaho.com/threads/187719-Classdef-not-found-error/



همانطور که مشاهده می‌کنید راه ذکر شده در این لینک را هم کنار بقیه موارد امتحان کردم تا به جواب نهایی پرسشگرِ این سوال رسیدم.

مطابق با آنچه در ویندوز من رخ داده برای ایشان هم رخ داده و هیچ راه‌حلی پیدا نشده. ایشان همین فایل‌ها را بدون تغییر در سیستم‌عامل مک امتحان نموده و بدون هیچ خطایی به پاسخ رسیدند.

لذا به نظر می‌رسد خطای موجود که در این قسمتِ سوال با آن برخوردم نه مربوط به من بلکه مربوط به یکی از نقطه ضعف‌های weka است که در بعضی سیستم‌عامل ها در خواندنِ برخی از کلاس‌ها که مربوط به فایل weka.jar است، مشکل ایجاد می‌کند.

لذا تنها راهی که برای پاسخ‌گویی به این سوال برایم باقی مانده است. ذکر مراحل آن است و از پیاده‌سازی آن هر چند زمان زیادی برای تصحیح فایل weka و کلاس آن گذاشتم، معذور شدم.

مراحلِ حل سوال:

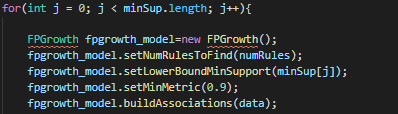
در ابتدا numrules را مطابق آنچه که در سوال گفته در فایل associationRulesMining.java مطابق با آنچه که در سوال گفته است تغییر می‌دهیم.



سپس minsup را به صورت یک آرایه تعریف می‌کنیم. دلیلِ این امر آن است که در یک لوپ الگوریتم‌ها را از نظر زمانی مقایسه می‌خواهیم کنیم.



*پارامتر‌ها در قسمتِ زیر تنظیم می‌شوند.*



همچنین همانطور که از قسمت قبلی به خاطر داریم، minMetric در setMinMetric به همان minconf اشاره می‌کند و لذا این مقدار را به 0.9 تغییر می‌دهیم.

همچنین همانطور که در بالا مشاهده می‌کنید minsup به صورت یک آرایه ظاهر شده است که هر بار یکی از درایه‌های این وکتور برای پردازش اعمال می‌شود.

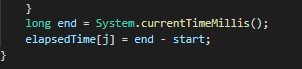
در نهایت با تعریفِ یک تایمر که به صورت یک وکتور و به اندازه‌ی minsup است، تایم هر اجرا را اندازه‌گیری می‌کنیم.



نقطه‌ی شروع به صورت زیر است.



و نقطه‌ی پایانِ اندازه‌گیری در هر iteration نیز به صورت زیر.



نهایتا برای هر یک از این دو الگوریتم پس از تعریفِ آنها این تایم‌ها را جداگانه ست کرده و با دستورِ put در یک فایلِ excel می‌ریزیم.

سپس با استفاده از ابزارِ مهندسیِ متلب فایل اکسل را به صورت زیر خوانده:



و در ماتریسِ A قرار می‌دهیم.

و در نهایت نیز با دستور plot در نرم‌افزار متلب آن را رسم می‌کنیم.

از اینکه مجبور هستم آخرین قسمت را به صورتِ ترکیبی از کد و توضیحی ارائه دهم عذر می‌خواهم. دلیلِ آن مشکل از سمت weka.jar بر روی برخی سیستم عامل‌ها از جمله سیستم‌عاملِ من است که عرض کردم.

با تشکر - بدیعی