

**دانشگاه صنعتی شیراز**

**دانشکده مهندسی کامپیوتر و فناوری اطلاعات گروه نرم‌افزار**

**گزارش سمینار کارشناسی‌ارشد**

**در رشته مهندسی کامپیوتر گرایش شبکه**

بررسی روش های استخراج معنی از داده های شبکه های اینترنت اشیا با استفاده از روش های یادگیری ماشین روی داده های بزرگ

**دانشجو:**

**سینا عطایی**

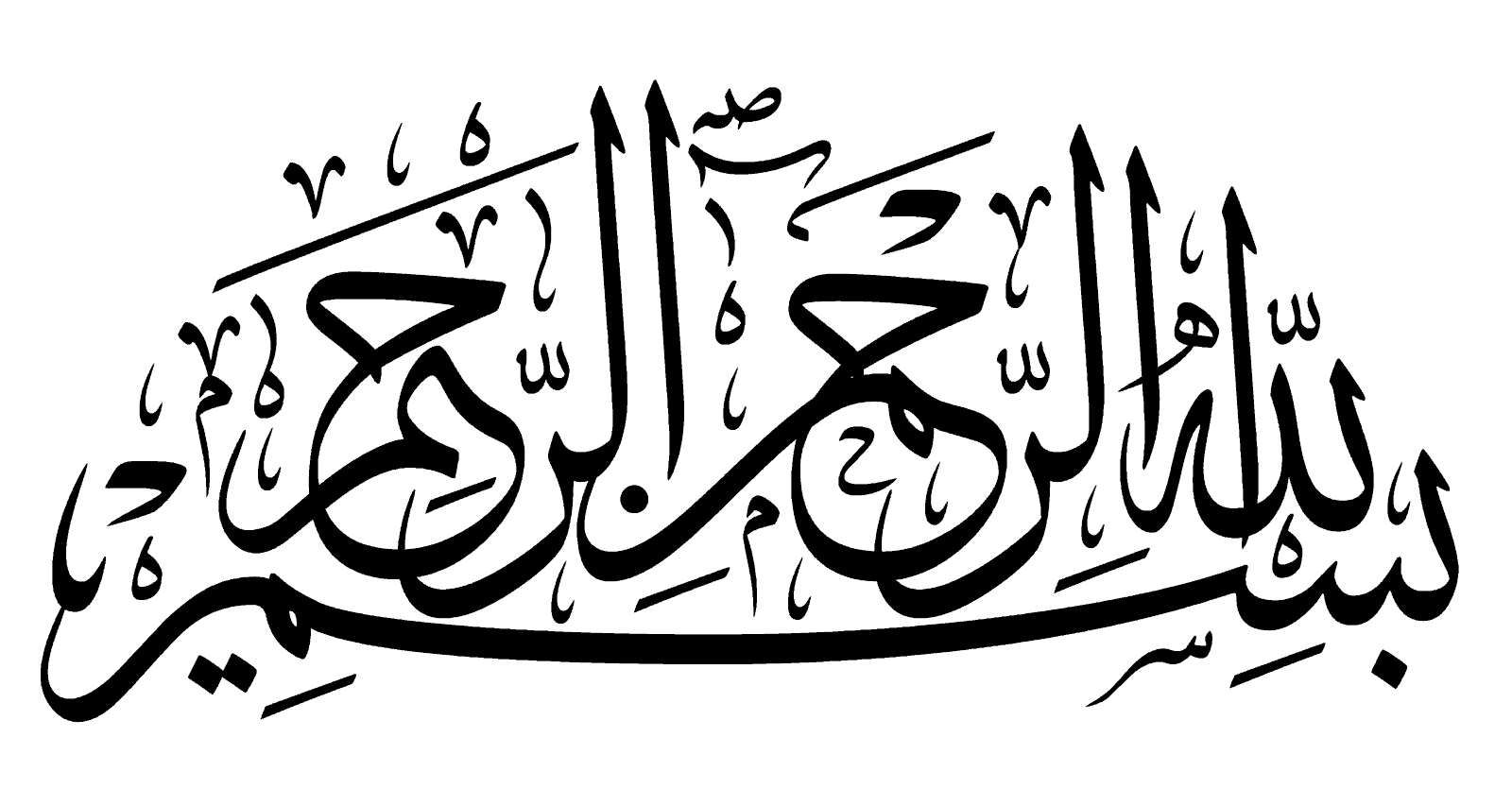
استاد راهنما:

دکتر جاویدان

استاد مشاور:

دکتر سرور

بهمن 97



چکیده

بررسی روش های استخراج معنی از داده های شبکه های اینترنت اشیا با استفاده از روش های یادگیری ماشین روی داده های بزرگ

نگارش :

از داده کاوی به عنوان یک ابزار قدرتمند در استخراج دانش از داده ها استفاده می شود. با توجه اینکه حجم وسیعی از اطلاعات در حال تولید شدن می باشد به داده کاوی برای تحلیل داده ها و کشف دانش موجود در آن ها نیاز خواهیم داشت. داده کاوی در زمینه های مختلفی از جمله پزشکی، مهندسی و اینترنت اشیاء کاربرد دارد. داده کاوی اینترنت اشیاء در سال های اخیر مورد توجه زیادی قرار گرفته است. در ابتدا مفاهیم مربوط به یادگیری ماشین، داده کاوی روی داده های بزرگ و کاربرد آن در حوزه اینترنت اشیاء را بررسی می کنیم و سپس برخی تحقیقات انجام شده در داده کاوی اینترنت اشیاء در سال های اخیر را مورد بررسی قرار می دهیم.

**کلمات کلیدی :** هوش مصنوعی، یادگیری ماشین، داده کاوی، اینترنت اشیاء، داده های بزرگ

فهرست مطالب

[فصل دوم: مروري بر ادبیات تحقیق 7](#_Toc536309894)

[1-1- مقدمه 8](#_Toc536309895)

[1-2- تعاریف و ادبیات پایه مربوط به داده کاوی 8](#_Toc536309896)

[1-3- داده کاوی با دادههای بزرگ 9](#_Toc536309897)

[1-4- روشهای برخورد با دادههای بزرگ 11](#_Toc536309898)

[1-4-1- روش MapReduce 11](#_Toc536309899)

[1-4-2- k نزدیک ترین همسایه 12](#_Toc536309900)

[1-5- نتیجه گیری 13](#_Toc536309901)

[فصل سوم: مروری بر تحقیقات اخیر انجام شده 14](#_Toc536309902)

[3-1- مقدمه 15](#_Toc536309903)

[3-2- مروری بر ادبیات موضوع 15](#_Toc536309904)

[3-3- نتیجه گیری 15](#_Toc536309905)

[مراجع 16](#_Toc536309906)

فهرست شکل‌ها

[شکل 1-1 دسته بندی روش های خوشه بندی 29](#ش4)

فهرست کلمات اختصاری

Iot Internet of Things

HDFS Hadoop Destributed File System

SAN Storage Area Network

فصل دوم: مروري بر ادبیات تحقیق

* 1. مقدمه

در این فصل ابتدا به تعریف داده کاوی پرداخته می­شود. سپس به صورت خاص‌تر، تعریف داده کاوی در IoT بیان می‌گردد.

* 1. تعاریف و ادبیات پایه مربوط به داده کاوی

داده کاوی[[1]](#footnote-1)چیست؟

در لغت به معنی بررسی و کاوش داده­ها می باشد.

از لحاظ مفهومی داده کاوی عبارت است از:

* ترکیب روش­های تحلیل داده­ها و الگوریتم­های پیشرفته برای پردازش داده­های با حجم بالا می باشد. داده کاوی امکان تحلیل نوع داده­های جدید و همچنین روش­های جدیدی برای تحلیل نوع داده­های قدیمی به ما می­دهد ]1[.
* فرآیندی است که به کمک کامپیوتر برای کاوش و تحلیل مجموعه­های عظیم داده­ها انجام می­شود تا مفهموم موجود در داده­ها را استخراج کند [2].
* داده کاوی فرایند استخراج داده­های مفید یا الگو از داده خام می­باشد ]6[.

فرض کنید یک مقدار مشخص از داده داریم، و به دنبال یک حالت مشخص از یک رویداد درون این داده­ها می­باشیم. حتی اگر داده­ها کاملا تصادفی باشند، می­شود انتظار داشت که این رویداد بخصوص رخ دهد. تعداد رویداد­ها با افزایش اندازه داده­ها، افزایش می یابد. اما این روی داد­­ها فریبنده هستند؛ زیرا داده­­های تصادفی همیشه تعدادی خصوصیت غیر عادی دارند که به نظر مهم می­آیند اما اینطور نیست. یک تئوری آماری به نام تصحیح بنفرونی[[2]](#footnote-2)، یک روش آماری ارائه می­دهد که میتواند اکثر رویداد­های فریبنده را تشخیص دهد. بدون ورود به جزئیات آماری، ما یک روش غیر­رسمی بر اساس اصل بنفرونی ارائه میکنیم، که کمک می­کند به اشتباه رویداد­های تصادفی را پردازش نکنیم. با فرض این­که داده­ها تصادفی هستند، تعداد رخ دادن رویداد مورد نظر را محاسبه می­کنیم. اگر عدد به دست آمده به شکل چشمگیری از تعداد واقعی نمومه­ها بیشتر باشد، آنگاه باید انتظار داشت هر چیزی که پیدا شده فریبنده باشد. البته این یک مسئله آماری ساختگی است تا یک مدرک برای پیدا کردن چیزی که به دنبال آن هستیم.

داده کاوی در اینترنت اشیاء[[3]](#footnote-3) جهت مدیریت مقدار زیاد داده­هایی که توسط دستگاه های اینترنت اشیاء تولید می­شوند، کاربرد دارد. داده کاوی شامل کشف و تجزیه و تحلیل دانش از مجموعه گسترده­ای از داده­ها است. هدف اصلی از استخراج داده، یافتن الگوهای مفید از مجموعه داده­های بزرگ دریافت شده از دستگاه­های اینترنت اشیاء (IoT) و سنسورها است. کشف دانش، تجزیه و تحلیل الگوها و برداشت اطلاعات، مفاهیمی هستند که در بحث داده کاوی در اینترنت اشیاء استفاده می­شوند. هدف اصلی از داده کاوی، ایجاد یک مدل کارآمد و توصیفی است که برای مجموعه داده مورد نظر مناسب باشد. بر اساس تعریف داده کاوی و عملکردهای آن، یک فرایند داده کاوی معمولا شامل این چهار مرحله است:

* جمع آوری داده
* آماده سازی داده­ها
* داده کاوی
* ارائه
  1. داده کاوی با داده­های بزرگ

در سیستم­های داده کاوی سنتی، فرایند استخراج نیاز به واحد­های محاسباتی پر­قدرت برای تجزیه و تحلیل داده­ها در سیستم­ها دارد. بنابراین، این پلت فرم محاسباتی با حداقل دو منبع، داده­ها و پردازنده­های سیستم­های کامپیوتری کار می­کند.

داده کاوی داده­های مقیاس کوچک فقط یک رایانه­ی رومیزی نیاز دارد و برای پردازش داده­ها از اندازه کافی نیز بیشتر است. در حالی که الگوریتم­های داده کاوی برای پردازش داده­های مقیاس متوسط طراحی شده­اند و شاید داده­ها بین چند سیستم توزیع شده باشند و بر روی حافظه یک سیستم جا نشوند. بنابراین، محاسبات یا استخراج موازی منقرض شد. الگوریتم­های داده کاوی معمول برای نمونه برداری و جمع آوری داده­ها از منابع مختلف سیستم باز­طراحی شده­اند و فرایند استخراج داده موازی را انجام می­دهند. به عنوان مثال، الگوریتم­های موازی K-means، الگوریتم طبقه بندی[[4]](#footnote-4) موازی و الگوریتم استخراج قوانین پیوستگی[[5]](#footnote-5) موازی برای پردازش داده های توزیع شده مورد استفاده قرار می گیرند.

هنگامی که مقیاس داده­ها فراتر از حد بزرگ است دیگر کامپیوتر­های رومیزی نمی­توانند جواب­گو باشند، بنابراین چارچوب داده کاوی از خوشه­های گره­های محاسباتی استفاده می­کند و با استفاده از برخی ابزارهای برنامه ریزی موازی مانند MapReduce مسائل را حل می­کند. MapReduce یک چارچوب برنامه نویسی برای تعداد زیادی از داده­ها است و از خوشه­های گره[[6]](#footnote-6) برای انجام پردازش موازی استفاده می­کند. داده­های بزرگ به طور مثال یعنی چند پتابایت یا ترابایت داده که به تعداد زیادی از وظایف یا کارهای کوچک تقسیم شده­اند و هر کدام از آن­ها در گره­های کامپیوتری قرار می­گیرند که خوشه­ها را تشکیل می­دهند و داده کاوی روی آن مجموعه­­­های بزرگ انجام می شود.

سیستم­های داده بزرگ در حال حاضر طراحی شده است و در برخی از صنایع موجود می­باشد که ترکیبی از اجزای سخت افزار و نرم­افزار است. بسیاری از شرکت­ها مانند آی­بی­ام، مایکروسافت، ترا دیتا و آمازون و غیره، محصولات خود را برای داده­های مشتریان منتشر کرده­اند تا با شناسایی روابط پنهان داده­های ذخیره شده به آن­ها خدمات رسانی و کمک کنند.

برای ایجاد پایگاه داده­های یادگیری هوشمند برای رسیدگی به پردازش داده­های بزرگ، نخستین و مهمترین گام این است که حجم داده­ها را فوق العاده زیاد افزایش داده و راه حل­هایی را برای ویژگی­های برجسته شده توسط قضیه HACE ارائه دهیم ]7[.

دیدگاه مفهومی چارچوب پردازش داده بزرگ را می توان با در نظر گرفتن درون داده ها به سه بخش طبقه بندی کرد:

1. دسترسی به داده ها و محاسبات.
2. حفظ حریم خصوصی داده ها و اطلاعات دامنه.
3. الگوریتم های داده کاوی داده های بزرگ ]8[.
   1. روش­های برخورد با داده­های بزرگ

به دلیل پیچیدگی محاسبات مجموعه داده­های بزرگ، حجم زیاد داده­ها مشکلات جدی برای الگوریتم­ها و تکنیک­های داده کاوی، یادگیری ماشین و غیره ایجاد کرده است. از این رو، حجم تحقیقات و محیط­های صنعتی به منظور پیدا کردن روش­ها و چارچوب­های جدید برای پردازش داده­های بزرگ، افزایش یافته است.

روش هایی که برای داده های بزرگ معرفی می شوند MapReduce، Apache Hadoop و Apache Spark است که عملکرد Hadoop را بهبود می بخشد [9].

* + 1. روش MapReduce

چارچوب MapReduce و الگوریتم­های آن برای پردازش داده­ها در مقیاس بزرگ و تعداد "n" خوشه توسعه داده شده است. MapReduce از دو عنصر هسته – mapper­ ها و reducer ها - در مدل برنامه­ نویسی استفاده می­کند. تابع map مجموعه­ای از جفت­های key/value موقت تولید می­کند و تابع reduce برای merge/combine تمام مقادیر متوسط key استفاده می­شود. ایده اصلی الگوریتم MapReduce این است که، هر گره کار map و reduce، مستقل از دیگر گره­های کار موازی است که از داده­ها و کلید مختلف برای انجام عملیات خود استفاده می کنند ]10[ ]11[.

توابع اصلی MapReduce عبارتند از Map، Combining / Shuffling و Reduce.

* داده ورودی: هنگامی که برنامه توسط یک گره خوشه Hadoop دریافت می­شود، گره برنامه را به قطعاتی به نام وظایف[[7]](#footnote-7) تقسیم می­کند و برنامه اجرا می­شود. در گره اصلی، هنگامی که کار دریافت شد، تقسیم شده و برای اجرای موازی با دیگر گره­های فرعی آماده می­شود. گره JobTracker برای ارتباط با Namenode استفاده می­شود تا Datanode متناظر آن را پیدا کند و وظایفش را ارسال کند.
* تابع Map: تابع map روی داده­های محلی هر گره یا گره کارگر عمل می­کند و داده­های خروجی را به صورت موقت در قالب جفت­های key/value -مانند (k1,v1)،(k2,v2)،...- ذخیره می­کند. گره اصلی نیز برای ترتیب و ترکیب تمام مقادیر key/value ها استفاده می­شود. بنابراین، فقط یک داده در هر واحد زمان پردازش می­شود.
* تابع Shuffle: خروجی تابع map به مرحله reduce می­رسد و key/value های جدید به همراه تمام داده­های مربوط به گره map به گره فعال اختصاص می­یابد. بنابراین، داده­ها با کلید یکسان به یک گره کارگر منتقل می­شوند.
* تابع Reduce: این تابع برای پردازش داده­های هر گره به صورت موازی و برای انجام عملیات مشخص reduce استفاده می­شود و تابع فقط یک بار برای هر مقدار key از مرحله map اجرا می­شود.
* داده خروجی نهایی: مدل MapReduce، تمام خروجی­های تابع reduce را ترکیب می­کند و با استفاده از key/value های مرتب شده در reduce، خروجی را برای کاربر تولید می­کند.

کاربرد MapReduce بسیار مفید است؛ اگر الگوریتم را به mapper ها و reducer هایی تقسیم کنیم تا با مجموعه داده­های بزرگ کار کنند. برخی از الگوریتم­های موازی می­توانند روی مجموعه داده­های کوچک کار کنند، اما نمی­توانند برای مجموعه داده­های بزرگ با استفاده از این روش MapReduce استفاده شوند ]12[ ]13[.

* + 1. ابزار آپاچی هادوپ[[8]](#footnote-8)

هادوپ به عنوان یک پروژه آپاچی در سطح بالا توسط بنیاد نرم افزار آپاچی ابداع و به زبان جاوا نوشته شده است. هادوپ به طور خاص طراحی شده است تا عملیات روی داده­ها در مقیاس بسیار بزرگ را انجام دهد و محیط محاسباتی آن بر اساس سیستم فایل خوشه­ای توزیع شده ساخته شده است. در هادوپ از MapReduce برای دستکاری داده­ها که در خوشه سرور ذخیره شده، استفاده می­شود که برنامه را به کارهای mapper و reducer تقسیم می­کند تا بتواند موازی سازی گسترده­ای را به وجود آورد ]14[.

هادوپ برای تعداد بیشتری از برنامه­های عملی و کاربردی موارد استفاده گسترده­تر و مجموعه داده­های بیشتری را فراهم آورده است. بر خلاف سیستم­های تراکنشی سنتی، هادوپ به منظور اسکن از طریق مجموعه داده­های بزرگ طراحی و ساخته شده و نتایجش را با استفاده از سیستم پردازش دسته­ای بسیار مقیاس پذیر و توزیع شده، تولید می­کند. از آنجایی که هادوپ حجم زیادی از داده را برای تجزیه و تحلیل پردازش می­کند، به عنوان یک تابع بر مدل داده­ای عمل می­کند و نه به عنوان یک داده بر مدل تابعی. هادوپ دارای دو بخش اساسی، فایل سیستم توزیع شده هادوپ[[9]](#footnote-9) و یک چارچوب برنامه ریزی MapReduce است.

* فایل سیستم توزیع شده هادوپ:

HDFS عمدتا مورد استفاده قرار می­گیرد تا خوشه­های هادوپ را به صدها و هزاران گره افزایش دهد. داده­های بزرگ در خوشه به تعدادی قطعه­های کوچک به نام بلوک تقسیم شده و بر روی دیگر گره­های خوشه توزیع می شود. به منظور افزایش مقیاس پذیری پردازش داده­های بزرگ، توابع mapper و reducer بر روی بلوک­های زیر مجموعه داده­ها اجرا می­شوند.

هادوپ معمولا از سرو­های موجود در یک خوشه بسیار بزرگ استفاده می­کند و این سرور دارای درایوهای حافظه داخلی ارزان است و برای پردازش داده­ها ذخیره شده در این سرورها از MapReduce استفاده می­شود. منطقه ذخیره سازی تحت شبکه[[10]](#footnote-10) برای محیط هادوپ استفاده می­شود؛ این امر ممکن است به دلیل سربار اضافی در ارتباطات شبکه، برای خوشه­های بزرگتر مشکل ایجاد کند. برای غلبه بر مشکل افزونگی داده­ها، هادوپ به خوشه اجازه می­دهد داده­ها را به تکه­های کوچکتر تقسیم کند و کارها را به طور صریح بر روی تمام سرورهای خوشه انجام دهد.

* گره Name:

نام گره یک سرور مخصوص برای مدیریت منطق قرار دادن داده در هادوپ است. این سرور تمام پرونده­های داده را در فایل سیستم توزیع شده ردیابی می­کند و اطلاعاتی مانند مکان ذخیره سازی بلوک را تولید می­کند. هر گونه دستکاری حافظه یا درخواست خواندن را با زمان پاسخ کم، انجام می­دهد. این سرور باید قوی­تر از سرورهای دیگر در خوشه هادوپ باشد تا تبدیل به گلوگاه سیستم نگردد.

* + 1. k نزدیک ترین همسایه [[11]](#footnote-11)

الگوريتم k نزدیک ترین همسایه بر اساس اين اصل که داده­هاي نزديک به هم در يک ديتاست داراي شباهت در خصوصيات مي­باشند کار مي کند. اگر نمونه­ها داراي برچسب باشند، داده­هاي بدون برچسب را مي­توان در کلاس نزديک­ترين داده به آن دسته بندي کرد. k نزدیک ترین همسایه، با بررسي k عنصر داده نزديک به داده مورد سوال آن داده را در بيشترين تکرار کلاس در k نمونه دسته­بندي مي­کند.

به صورت عمومي، هر نمونه را مي­توان يک نقطه در يک فضاي n بعدي در نظر گرفت که هر کدام از ابعاد توصيف کننده مقدار يکي از n ويژگي­هاي نمونه مي­باشد. مکان هر نقطه در فضا به اندازه فاصله نسبي آن از ديگر نقاط داراي اهميت نمي باشد. هدف کمينه سازي فاصله ميان نمونه­هاي يک کلاس و بيشينه سازي فاصله ما بين نمونه­هاي موجود در کلاس­هاي مختلف است.

قدرت روش k نزدیک ترین همسایه در کاربردهاي واقعي بسياري اثبات شده است اما استثناهايي در رابطه با کارکرد مناسب k نزدیک ترین همسایه وجود دارد :

- روش k نزدیک ترین همسایه به فضاي ذخيره سازي زيادي نياز دارد.

- اين روش به تابع تشخيص شباهتي که استفاده شده حساس است.

- مشکل در تعيين مقدار مناسب براي k

- هزينه محاسباتي بالا

اوکاموتو و یوگامی دقت مورد انتظار k نزدیک ترین همسایه را به عنوان تابعي از ويژگي­هاي دامين مورد استفاده مانند تعداد نمونه­هاي موجود در مجموعه آموزش و تعداد خصوصيات مرتبط و نامرتبط، نرخ نويز، احتمال مربوط به هر خصوصيت و مقدار k نشان داده اند [3].

همانگونه که در قبل اشاره شد يکي از معايب روش­هاي مبتني بر نمونه ميزان هزينه محاسباتي بالا در آن­ها مي­باشد. يک مسئله کليدي در بسياري از کاربردها تشخيص ويژگي­هايي که در مدل سازي بايد استفاده شود توسط روش­هاي انتخاب ويژگي تعيين مي­شود [4].

انتخاب ويژگي مي­تواند به بهبود دقت و مقياس پذيري و همچنين کاهش زمان دسته­بندي کمک کند. مساله ديگر انتخاب نمونه­هايي است که براي مدل سازي بايد استفاده شوند [5].

* 1. نتیجه گیری

فصل سوم: مروری بر تحقیقات اخیر انجام شده

3-1- مقدمه

در فصل قبل به چالش­های که در داده کاوی وجود دارد پرداخته شد. در این فصل به تحقیق­های جدیدی که در این زمینه انجام داده شده است، پرداخته خواهد شد.

3-2- مروری بر ادبیات موضوع

3-3- نتیجه گیری

در این فصل به تحقیقات جدید انجام شده در زمینه داده کاوی پزشکی پرداخته شد. با بررسی کارهای انجام شده در این زمینه می توان دریافت که در سال های اخیر به روش های ترکیبی داده کاوی از جمله شبکه عصبی فازی، دسته بندی مشارکتی، دسته بندی تلفیقی و... توجه بیشتری شده است. برخی روش ها مانند ماشین بردار پشتیبان کاربرد بیشتری در داده کاوی پزشکی داشته و نتایج بهتری را نسبت به سایر روش ها ارائه می دهد

مراجع

مراجع

[1] Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. *Introduction To Data Mining*. Boston: Pearson Addison Wesley, 2005.

[2] S. Sumathi and S.N. Sivanandam: *Data Warehousing, Data Mining, and Olap*, Studies in Computational Intelligence (SCI) **29**, 21–73 (2006).

[3] N. Y. Seishi Okamoto, “Effects of domain characteristics on instance-based learning algrithims,” *Theor. Comput Sci*, vol. 298, pp. 207–233, 2003.

[4] L. Yu and H. Liu, “Efficient Feature Selection via Analysis of Relevance and Redundancy,” *J. Mach. Learn. Res.*, vol. 5, pp. 1205–1224, Dec. 2004.

[5] J. S. Sánchez, R. Barandela, and F. J. Ferri, “On filtering the training prototypes in nearest neighbour classification,” in *Topics in Artificial Intelligence*, Springer, 2002, pp. 239–248.

[6] Shobanadevi, A., & Maragatham, G. *Data mining techniques for IoT and big data - A survey*. In Proceedings of the International Conference on Intelligent Sustainable Systems, ICISS 2017 pp. 607–610. (2018).

[7] S. Rathee, M. Kaul, and A. Kashyap, “R-Apriori: An Efficient Apriori based Algorithm on Spark,” Acm, pp. 27–34, 2015.

[8] X. Wu, X. Zhu, G. Q. Wu, and W. Ding, “Data mining with big data,” IEEE Trans. Knowl. Data Eng., vol. 26, no. 1, pp. 97–107, 2014.

[9] Shen Bin, Liu Yuan, and Wang Xiaoyi, “Research on data mining models for the internet of things,” 2010 Int. Conf. Image Anal. Signal Process., pp. 127–132, 2010.

[10] M. Marjani et al., “Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges,” IEEE Access, vol. PP, no. 99, p. 1, 2017.

[11] G. Bello-Orgaz, J. J. Jung, and D. Camacho, “Social big data: Recent achievements and new challenges,” Inf. Fusion, vol. 28, pp. 45–59, 2016.

[12] J. Archenaa and E. A. M. Anita, “A survey of big data analytics in healthcare and government,” *Procedia Comput. Sci.*, vol. 50, pp. 408–413, 2015.

[13] J. L. Reyes-Ortiz, L. Oneto, and D. Anguita, “Big data analytics in the cloud: Spark on Hadoop vs MPI/OpenMP on Beowulf,” *Procedia Comput. Sci.*, vol. 53, no. 1, pp. 121–130, 2015.

[14] A. Gandomi and M. Haider, “Beyond the hype: Big data concepts, methods, and analytics,” *Int. J. Inf. Manage.*, vol. 35, no. 2, pp. 137–144, 2015.

1. Data Mining [↑](#footnote-ref-1)
2. Bonferroni correction [↑](#footnote-ref-2)
3. Internet of Things [↑](#footnote-ref-3)
4. Classification [↑](#footnote-ref-4)
5. Association rule mining [↑](#footnote-ref-5)
6. Clusters of nodes [↑](#footnote-ref-6)
7. Tasks [↑](#footnote-ref-7)
8. Apache Hadoop [↑](#footnote-ref-8)
9. HDFS [↑](#footnote-ref-9)
10. SAN [↑](#footnote-ref-10)
11. K-Nearest Neighbor [↑](#footnote-ref-11)