UNIVERSIDAD NACIONAL DE LA AMAZONÍA PERUANA

FACULTAD DE INGENIERÍA
DE SISTEMAS E INFORMÁTICA

NUEVOS ALGORITMOS PARA EL CALCULO DE FUNCIONES MATRICIALES

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO DE SISTEMAS E INFORMÁTICA

Presentado por el Bachiller:
:
MAYCOL ALCALDE MONDACA

ASESOR: Mag. CARLOS ALBERTO GARCIA CORTEGANO

IQUITOS – PERÚ

2017

MIEMBROS DEL JURADO

Ing. Alejandro Reátegui Pezo
Presidente

Ing. Jimmy Max Ramirez Villacorta
Primer Miembro

Ing. Grecia Milagros Barrera Ortiz
Segundo Miembro

Ing. Carlos Alberto García Cortegano
Asesor

UNIVERSIDAD NACIONAL DE LA AMAZONIA PERUANA
FACULTAD DE INGENIERIA DE SISTEMAS E INFORMATICA

ACTA DE SUSTENTACION DE TESIS

Siendo las 19:00 horas del día 29 de AGOSTO del 2017, en la Instalación del Auditorio de esta Facultad, se ha constituido el jurado examinador integrado por los siguientes miembros:

Presidente : Ing. Alejandro Reátegui Pezo
Primer Miembro : Ing. Jimmy Max Ramírez Villacorta
Segundo Miembro : Ing. Grecia Milagros Barrera Ortiz

Acto seguido, se recibió al Tesista: **Maycol Alcalde Mondaca,** quien sustento el tema de Tesis: **"Nuevos Algoritmos Para El Cálculo De Funciones Matriciales"**, para optar el Título Profesional de Ingeniero de Sistema e Informática, de acuerdo a lo establecido en el Reglamento de Grados y Títulos de la Facultad de Ingeniería de Sistemas e Informática.

Posteriormente, al Acto de sustentación de la Tesis, se procedió al cálculo de Calificación obteniéndose el siguiente resultado:

	Calificaciones	
	En número	**En letras**
La Ejecución de la Tesis	17.6	DIECISIETE y 60/100
La Sustentación de la Tesis.	17.5	DIECISIETE y 50/100
Promedio Final	17.6	DIECISIETE y 60/100

De la evaluación se desprende que el resultado es:

APROBADO

Con las siguientes observaciones:

Se da por concluido el acto de sustentación a las 20:00 horas, procediéndose a firmar el Acta .

Ing. Alejandro Reátegui Pezo
Presidente

Ing. Jimmy Max Ramírez Villacorta
Primer Miembro

Ing. Grecia Milagros Barrera Ortiz
Segundo Miembro

RESUMEN

La investigación tuvo como objetivo Desarrollar algoritmos computacionales para los métodos y técnicas estadísticas más usados para estudiantes de la Facultad de ingeniería de sistemas e informática de la universidad Nacional de la Amazonia Peruana., la investigación pertenece al tipo básico con nivel descriptivo y el diseño fue el no experimental de tipo transversal, cuyos resultados se presentan los algoritmos estadísticos, La muestra para la presente investigación estuvo constituida por 60 estudiantes de la Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonia Peruana, la selección de la muestra se efectuó en forma no probabilística intencionada.

.

La técnica empleada en la recolección de los datos fue Investigación documental. La hoja de codificación y el instrumento fue Análisis de contenido, Concluyendo: Los algoritmos estadísticos cumplen la función de instruir a los usuarios en el manejo de procesos estadísticos, de los cuales el estudiante podrá tener una mejor idea de los procedimientos matemáticos necesarios para resolver los distintos postulados estadísticos.

ABSTRACT

Research as was objective To develop computer algorithms for the most used statistical methods and techniques for students of the School of Systems Engineering and Computer Science of the National University of the Peruvian Amazon.

, the research belongs to the basic type with a descriptive level and the design was the non-experimental transversal type, whose results are presented by the statistical algorithms. The sample for the present investigation consisted of 60 students from the Faculty of Systems Engineering and Computer Science. The National University of the Peruvian Amazon, the selection of the sample was carried out in an intentional non-probabilistic manner.

.

The technique used in the data collection was documentary research. The coding sheet and the instrument was content analysis, concluding: The statistical algorithms fulfill the function of instructing the users in the handling of statistical processes, of which the student will have a better idea of the mathematical procedures necessary to solve the different statistical postulates.

DEDICATORIA

- A mis padres, por su amor y apoyo incondicional en todo momento.
- A los lectores de este documento.
- A usuarios del producto desarrollado.

AGRADECIMIENTOS

- A mi Dios, por iluminarme mi camino en todo momento, gracias por darme fuerzas para seguir adelante y permitirme compartir mi vida con seres tan especiales.

- A toda mi familia, por el amor brindado, experiencias y consejos dados sin esperar nada a cambio, los amo.

- A la ilustre Universidad Nacional de la Amazonia Peruana, por ser la casa de estudios que me brindo sus conocimientos durante toda mi carrera.

- A la Facultad de Sistemas e Informática, su plana docente y administrativo, por brindarme la formación académica y profesional que poseo.

- A las personas relacionadas con la bibliografía de este documento y las fuentes a las que acudí para realizar este trabajo.

- Y a todas aquellas personas que de una u otra manera me ayudaron en la elaboración de este trabajo, así como en la culminación de mi carrera profesional.

ÍNDICE DE CONTENIDO

Resumen…….. ii

Abstract……. iii

Introducción…….. 01

Capítulo I: Planteamiento del Problema…….. 03

Objetivos……... 04

Justificación…….. 05

Capítulo II: Marco Teórico…….. 06

Antecedentes……. 06

Planteamiento Teórico……. 07

Hipótesis……. 27

Capítulo III: Metodología……. 28

Capítulo IV: Resultado y Discusiones…….. 30

Capítulo V: Conclusiones……. 40

Capítulo VI: Recomendaciones…….. 41

Capítulo VII: Referencias Bibliográficas……. 42

Anexos:……. 44

INTRODUCCION

En la Escuela Profesional de Ingeniería de sistemas e Informática de la Universidad Nacional de la Amazonia Peruana es habitual que un estudiante se vea limitado en su quehacer académico por los programas como SPSS, Minitab, SAS, NCSS, etc. Estas herramientas se usan a nivel de usuario cuando la necesidad de muchos estudiantes es entender algoritmos computacionales con código fuente visible aplicados a los cursos de estadística. En las asignaturas de estadísticas se estudian en su mayoría la teoría, conceptos, teoremas, funciones estadísticas, métodos y técnicas, cuya aplicación se hace mediante el uso de cálculos científicos de dicho análisis estadístico, en las asignaturas de informática se da una formación netamente centrada en el área, creando así un divorcio involuntario entre ambas asignaturas, siendo necesario herramientas de software libre que facilite el estudio coordinado integrando.

En la actualidad las asignaturas de estadística se imparte base teórica, pero no se desarrollan algoritmos computacionales de su contenido en el área de informática ocurre de manera similar, se dictan cursos taller de programación centrada en algoritmos genéricos omitiendo algoritmos de estadística aplicada, cuando en la actualidad la estadística constituye técnicas y métodos de optimización, complejidad y eficiencia de los algoritmos computacionales, los cuales deben ser implementados en un software aplicado en las asignaturas para comprender en las técnicas y métodos, es importante generar algoritmos computacionales y más adelante sea implementado el código fuente como parte del taller de las asignaturas de informática.

Para el estudio de algoritmos estadísticos existe en Internet código fuente limitado, superficial y disperso, que no permite ser analizado adecuadamente variando en metodología, lenguaje en el que fue escrito y la documentación de diverso tipo; por otro lado tenemos la existencia de código fuente de calidad en programas avanzados como R ó PSPP, cuyos comentarios no están en nuestro idioma, sus algoritmos son muy refinados siendo orientados para su aplicación en ambientes científicos de primer nivel lo que dificulta su entendimiento para un estudiante de pregrado, es decir están al alcance del entendimiento de muy pocos estudiantes de la escuela profesional con avanzado conocimiento de programación.

Considerando estos factores es necesario generar algoritmos computacionales sencillos y eficientes para cada una de las técnicas y métodos estadísticos estudiados

en las aulas, de manera que estas sean entendidas y se puedan implementar en programas.

La investigación comprende los siguientes capítulos:

- ✓ Capítulo I: Planteamiento del problema
- ✓ Capitulo II: Marco Teórico que abarca: antecedentes del estudio, Planteamientos teóricos; Justificación.
- ✓ Capitulo III: Metodología
- ✓ Capitulo IV: Resultado y discusión.
- ✓ Capitulo V: Conclusiones
- ✓ Capitulo VI: Recomendaciones
- ✓ Capitulo VII: Bibliografía
- ✓ Anexos

CAPITULO I: PLANTEAMIENTO DEL PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

En la actualidad las asignaturas de estadística se imparte base teórica, pero no se desarrollan algoritmos computacionales de dicha base teórica como practica de su implementación en las asignaturas del área de informática ocurre de manera similar, se dictan cursos taller de programación centrada en algoritmos genéricos omitiendo algoritmos de estadística aplicada, cuando en la actualidad la estadística constituye técnicas y métodos de optimización, complejidad y eficiencia de los algoritmos computacionales, que muy bien deben ser implementados como software aplicado en las asignaturas para comprender la aplicabilidad de dichas técnicas y métodos, es importante generar algoritmos computacionales y más adelante sea implementado en código fuente como parte del taller de las asignaturas de lenguajes de Programación.

Para el estudio de algoritmos estadísticos existe en Internet código fuente limitado, superficial y disperso, que no permite ser analizado adecuadamente variando en metodología, lenguaje en el que fue escrito y la documentación de diverso tipo; por otro lado tenemos la existencia de código fuente de calidad en programas avanzados como R ó PSPP, cuyos comentarios no están en el idioma español, sus algoritmos son muy refinados siendo orientados para su aplicación en ambientes científicos de primer nivel lo que dificulta su entendimiento para un estudiante de Ingeniería de sistemas e informática.

Considerando estos factores es necesario generar algoritmos computacionales sencillos y eficientes para cada una de las áreas de estadística estudiadas en las aulas de las diferentes asignaturas del área de estadística, de manera que estas sean implementadas y automatizadas en las asignaturas de informática, además pueda constituir la generación de software aplicado

1.1. Problema General

¿De qué manera es posible desarrollar la integración de base teórica de la estadística en algoritmos computacionales?

1.2. OBJETIVOS

1.2.1. OBJETIVO GENERAL

Desarrollar algoritmos computacionales para los métodos y técnicas estadísticas más usados para estudiantes de la Facultad de ingeniería de sistemas e informática de la universidad Nacional de la Amazonia Peruana.

1.2.2. OBJETIVOS ESPECIFICOS

- Diseñar algoritmos computacionales para los métodos y técnicas estadísticas más usados.
- Construir módulos de código abierto de los algoritmos generados en un lenguaje de programación.
- Integrar diferentes módulos con características de fácil uso y programación amigable.

1.3. JUSTIFICACION

En la actualidad las asignaturas de estadística se imparte base teórica, pero no se desarrollan algoritmos computacionales de su contenido en el área de informática ocurre de manera similar, se dictan cursos taller de programación centrada en algoritmos genéricos omitiendo algoritmos de estadística aplicada, cuando en la actualidad la estadística constituye técnicas y métodos de optimización, complejidad y eficiencia de los algoritmos computacionales, los cuales deben ser implementados en un software aplicado en las asignaturas para comprender en las técnicas y métodos, es importante generar algoritmos computacionales y más adelante sea implementado el código fuente como parte del taller de las asignaturas de informática.

Para el estudio de algoritmos estadísticos existe en Internet código fuente limitado, superficial y disperso, que no permite ser analizado adecuadamente variando en metodología, lenguaje en el que fue escrito y la documentación de diverso tipo; por otro lado tenemos la existencia de código fuente de calidad en programas avanzados como R ó PSPP, cuyos comentarios no están en nuestro idioma, sus algoritmos son muy refinados siendo orientados para su aplicación en ambientes científicos de primer nivel lo que dificulta su entendimiento para un estudiante de pregrado, es decir están al alcance del entendimiento de muy pocos estudiantes de la escuela profesional con avanzado conocimiento de programación.

CAPITULO II: MARCO TEORICO

2.1. ANTECEDENTES

Jiménez, (2003) se ha pretendido impulsar el acercamiento de los profesionales e investigadores de la estadística a esta nueva manera de entender el desarrollo de software, haciéndolo además de una manera activa. Para ello, se ha trabajado en la construcción de un framework específico para el modelado estadístico con MLG (Modelo Lineal General).

Javier Ibañez Gonzales (2006). La tesis se centra, fundamentalmente, en el desarrollo de métodos y la implementación de algoritmos que calculan Funciones de Matrices, y su aplicación a la resolución de Ecuaciones Diferenciales Ordinarias (EDOs) y Ecuaciones Diferenciales Matriciales de Riccati (EDMRs), con estos métodos se refiere a cualquier software de cálculo matemático requiere de algoritmos y fundamentalmente base teórica del tema.

César Augusto Cayetano Muro, Koening Alfredo Rojas Bustillos (2007) StadCore - Herramienta Estadística Modular: Una Orientación a los Procesos de Confiabilidad y Análisis de Psicometría, Las herramientas orientadas a áreas específicas de estudio son prácticamente escasas y normalmente requieren del uso de más de una aplicación para llegar a un resultado. De esta forma, se obtienen soluciones parciales a una problemática, lo que ocasiona la necesidad de recurrir a otras herramientas para que terminen el trabajo. Un caso muy particular de este problema es el del proceso de psicometría en el cual las etapas de confiabilidad y análisis requieren del uso de varias herramientas. Además, las soluciones obtenidas en cada etapa no son completas dado que se necesita realizar procesos no automatizados que tienen mayor probabilidad de conllevar a errores. La presencia de estos errores en un test trae consigo importantes consecuencias dado que podría obtenerse conclusiones inexactas. Para solucionar este problema se implementó una herramienta que sea lo suficientemente flexible como para permitir ampliar sus funcionalidades a diversas áreas de trabajo, debido a que las hojas de cálculo electrónicas son aplicaciones que fácilmente pueden desempeñarse en cualquier área de estudio, se utilizó esta característica como interfaz de la herramienta.

2.2. PLANTEAMIENTOS TEORICOS

2.2.1. ANÁLISIS DESCRIPTIVO

Según Galán M. (2003), existen dos tipos de fenómenos en estadística descriptiva que son los fenómenos deterministas y los no deterministas o aleatorios.

a) Fenómeno aleatorio. Las experiencias que al ser repetidas en condiciones que estimamos similares, nos llevan a resultados diferentes constituyen los llamados fenómenos aleatorios.

Tabulación de frecuencias

Según Galán M. Es una distribución organizada de los datos recogidos en un estudio. Contiene un listado de distintas modalidades del fenómeno considerado, con la frecuencia absoluta, relativa y acumulada de cada una. Cuando el número de modalidades es demasiado grande (eso ocurre siempre con modalidades continuas) se agrupan en clases.

Reglas prácticas.

- Las clases han de ser excluyentes.
- Los límites de cada clase deben tener más precisión que las medidas realizadas.
- Aunque no tiene que ser necesariamente así, es conveniente que la amplitud de los intervalos sea constante.
- Todos los datos de una clase quedan representados por la marca de clase, que es el valor medio de intervalo que forma la clase. De esta manera todos los cálculos se realizan como si en lugar de tener N valores

Según Mitacc. M. la regla general para la construcción de una distribución de frecuencias de una variable agrupada en intervalos es:

1. Determinar el rango. Es conveniente determinar el rango de los datos obtenidos, los cuales se han definido por:

R= max {X} – min {X}, para construir intervalos traslapados o

R= max {X} – min {X} + unidad de medida para construir intervalos no traslapados

2. Determinación del número de intervalos de clases. Consiste en dividir el rango en un número conveniente de intervalos de clase, generalmente del mismo tamaño, es conveniente tener entre 5 y 20 intervalos de clase. Entre más datos se tengan más

intervalos de clase deben considerarse. No hay una fórmula exacta para calcular el número de intervalos de clase.

Una solución de referencia podría ser.

a. Número de clases = *k* = *5*, si n tamaño de la muestra y $k = \sqrt{n}$, si n = 25.
b. Fórmula de Sturges: $k = 1 + 3.22l$ [illegible] $_1$ (n)

Cuando los resultados para obtener *k* en a) y b) son números reales con decimales, entonces se redundará al entero inmediato mayor.

3. Determinación del tamaño de intervalos. Como regla general para encontrar la longitud de los intervalos y último intervalo. Esto es para permitir uniformidad en las comparaciones de frecuencias de clase.

4. Determinación de los límites de clase. Se debe tomar los resultados numéricos más bajos de los datos originales como el límite inferior del primer intervalo de clase. Agregar C para obtener el límite superior de dicha clase, esto es, cuando se usa

Rango (agregar 0-1 para obtener el límite superior de clase, caso que se usa intervalos no traslapados). Se añade los demás intervalos.

Rango= R= max {X} – min {X}, (agregar 0 -1 para obtener el límite superior)

5. Determinación de la frecuencia de clase. Consiste en determinar el número de observaciones que caen en cada intervalo de clase.

Representación gráfica de variables cuantitativas. Las más usadas son:

1. Diagrama de barras. Esta forma de distribución gráfica es propia de las distribuciones que tienen muchas observaciones, pero pocos valores distintos de una variable.

Dicho diagrama se elabora colocando en el eje de las abscisas los distintos valores de la variable y sobre cada una de ellas se levanta una línea perpendicular, cuya altura es la frecuencia (absoluta o relativa) de dicho valor.

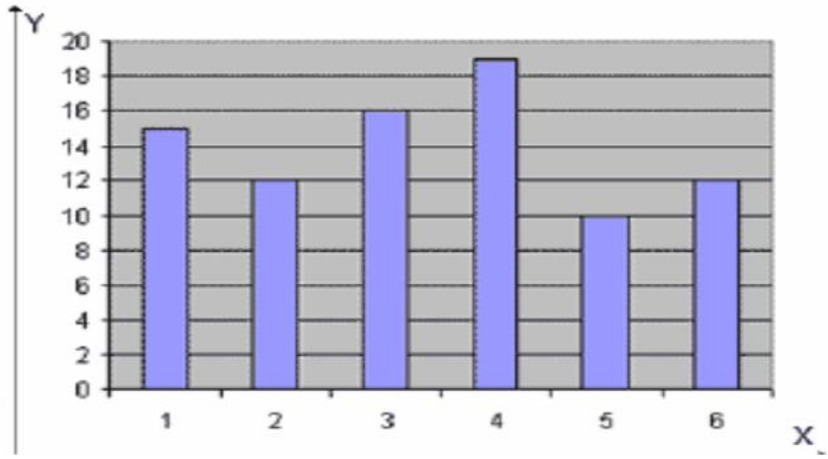


Figura 1. Diagrama de barras

2. Histograma. Es una representación gráfica de frecuencias agrupadas en intervalos de clase, mediante una serie de rectángulos contiguos que tienen:

a. Sus bases sobre un eje horizontal (eje de las X) con centros en las marcas de clase y longitud igual al tamaño de los intervalos de clase.

b. Las alturas proporcionales a la frecuencia (absoluta o relativa) tomamos sobre el eje de las Y.

A veces conviene más graficar en el histograma las frecuencias relativas en lugar de las frecuencias absolutas. En este caso, la altura correspondiente a cada rectángulo que habrá que levantar sobre el eje de las ordenadas será el cociente entre la frecuencia relativa del mismo y la amplitud del intervalo. El único cuidado que debe tenerse es que el área total del histograma sea igual a la correspondiente suma total de áreas de cada rectángulo.

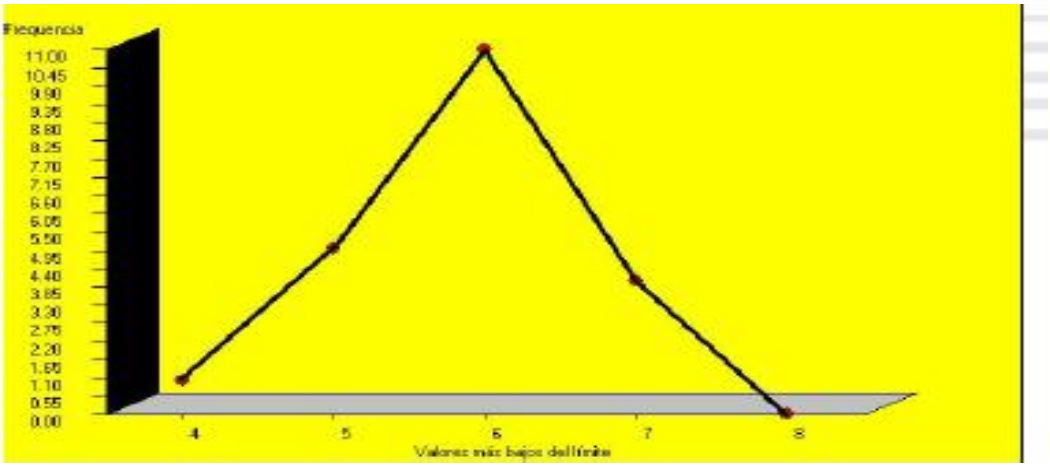


Figura 2. Histogramas

3. Polígono de frecuencias. Si la variable está agrupada en intervalos de clase, el polígono de frecuencias se obtiene uniendo los puntos medios de las bases superiores del rectángulo del histograma.

4. Polígono de frecuencia acumulada u ojiva. Esta representación es válida para variables estadísticas agrupadas en intervalos de clase.

En el eje de las abscisas representamos los distintos intervalos de clase que han de estar naturalmente traslapados. En el extremo superior de cada intervalo se levanta una vertical con altura igual a la frecuencia (absoluta o relativa) acumulada, luego se unen los extremos superiores de las verticales con segmentos rectilíneos. Así el polígono de frecuencias absoluta acumuladas alcanzara su máxima altura en el último intervalo.

Tabla de Frecuencia

Horas	ni	Ni
[46,52]	8	8
(52,58]	6	14
(58,64]	4	18
(64,70]	6	24
(70,76]	3	27
(76,82]	3	30

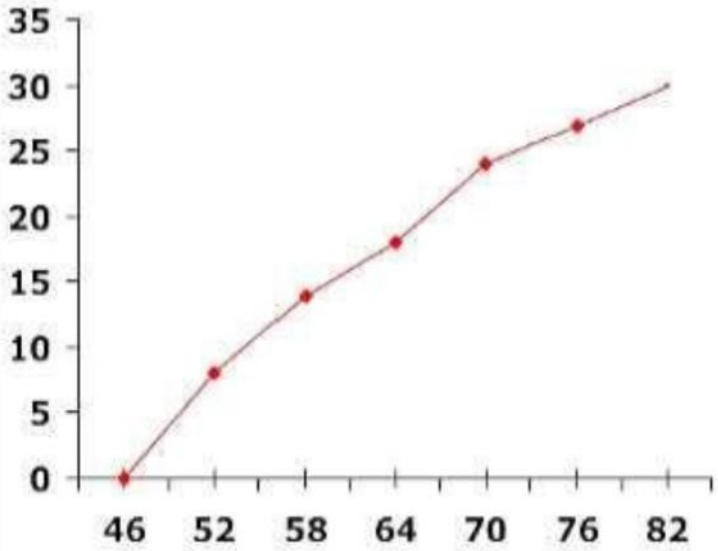


Figura 3. Frecuencia Acumulada u ojiva.

Medidas de la tendencia central

Mittac M. (1996), indica que una medida de tendencia central es un índice de localización central empleado en la descripción de las distribuciones de frecuencias. También sirve como una base para medir y evaluar valores anormalmente altos o anormalmente bajos (o valores extremos).

Características del valor central.

1. Debe estar definido en forma objetiva.

2. Debe depender de toda la información obtenida en lo posible.

3. Debe ser fácil de comprender (no debe tener un carácter de abstracción) y de interpretar.

4. Debe ser fácil de calcular.

5. Debe ser estable (no debe ser sensible a fluctuaciones).

6. Debe ser adecuado a cálculos algebraicos posteriores.

Los promedios para datos agrupados más usuales son:

a. **La Media aritmética o media.** Según Mitacc (1996). Sean $x1, x2, ..., xk$ valores de la variable X ponderamos por sus respectivas frecuencias absolutas $f_1, ... f_k$. La media de la variable X es dada por.

$$M(X) = \overline{X} = \frac{\sum_{i=1}^{n} f_i x_i}{n}$$

D $n = \sum_{i=1}^{n} f_i$, y $h_1, \ldots, h_k$ son las frecuencias relativas respectivas

b. **La mediana.** Es un valor que divide a un conjunto de observaciones ordenadas en forma ascendente o descendente en dos grupos de igual número de observaciones.

La formula de la mediana para datos agrupados es:

$$\overline{X} = l_m + \left[\frac{\frac{n}{2} - F_{K-1}}{F_k - F_{k-1}}\right] . C_m$$

donde

ℓ_{med}: Límite inferior de la clase que contiene a la mediana.

n : Tamaño de muestra

C_{med}: Amplitud de clase que contiene a la mediana.

F_k: Frecuencia acumulada de la clase que contiene a la Mediana.

Medidas de Dispersión o Concentración

Según Mitacc M. Las medidas de dispersión son los que cuantifican el grado de concentración o de dispersión de los valores de la variable en torno de un promedio o valor central de la distribución. Las medidas de dispersión se necesitan para dos propósitos básicos:

- Para verificar la confiabilidad de los promedios.
- Para que sirva como base para el control de la variación misma.

También podemos decir que los términos *concentración* y *dispersión* pueden ser utilizados indistintamente, pues se da la relación.

Alta dispersión -> Baja concentración

Baja dispersión Alta concentración.

Las medidas de dispersión que se utilizan con mayor frecuencia son:

1. Recorrido o rango.

El recorrido de una variable estadística es simplemente la diferencia entre su valor máximo y su valor mínimo y se denota por:

R= max {X} – min {X},

1. Recorrido intercuartílico.

Como los cuartiles son tres puntos (valores) que dividen un ordenamiento de datos o una distribución de frecuencias en 4 grupos aproximadamente.

Entonces la medida dada por.

$$Q_1 = Q_3 - Q_1$$

Llamada recorrido o amplitud intercuartílico, incluye la mitad central de los valores.

2. Recorrido semi-intercuantílico.

Esta medida se basa en la posición ocupada por el 50% de los valores centrales de la distribución, es dada por:

$$Q_{s1} = \frac{Q_3 - Q_1}{2}$$

3. Desviación media.

La desviación media o promedio, es simplemente la media aritmética de los valores absolutos de las desviaciones de todos los valores en relación con algún punto central, tal como la media o la mediana. Formalmente tenemos la siguiente definición.

Sean $x_1, x_2, \ldots, x_k$ valores de la variable x, con frecuencias absolutas $f_1, \ldots, f_k$, respectivamente. La desviación media de X con respecto a un promedio p es dado por

$$D_M(p) = \frac{\sum_{i=1}^{k} f_i \, |x_i - p|}{n} \quad d \qquad n = \sum_{i=1}^{k} f_i$$

La fórmula para la desviación con respecto a la media aritmética es:

$$D_M(\bar{X}) = \frac{\sum_{i=1}^{k} f_i \, |x_i - \bar{X}|}{n}$$

Y con respecto a la mediana es

$$D_M(\bar{X}) = \frac{\sum_{i=1}^{k} f_i \, |x_i - \bar{X}|}{n} \quad 0 \quad D_M(\bar{X}) = \sum_{i=1}^{k} h_i \; |x_i - \bar{X}|$$

Donde $h_1, h_2, \ldots, h_k$ son las frecuencias relativas.

5. Varianza y desviación estándar.

Sean $x_1, x_2, \ldots, x_k$ valores de la variable X, con frecuencias absolutas $f_1, \ldots, f_k$, respectivamente la varianza muestral de X es dada por:

$$Var[X] = S^2 = \frac{\sum_{i=1}^{k} f_i (x_i - \overline{X})^2}{n-1} = \frac{n}{n-1} \sum h_i (x_i - \overline{X})^2$$

Donde: $n = \sum_{i=1}^{k} f_i$ y $h_1, \ldots, h_k$ son las frecuencias relativas.

La varianza poblacional se define en términos de la media poblacional μ, esto es.

$$\sigma^2 = \frac{\sum_{i=1}^{k} f_i \, (x_i - \mu)^2}{N}$$

Donde $N = \sum_{i=1}^{k} f_i$ es el tamaño de la población

2.2.2. ANÁLISIS COMPARATIVO

A la teoría de pequeñas muestras también se le llama teoría exacta del muestreo, ya que también la podemos utilizar con muestras aleatorias de tamaño grande.

Veremos un nuevo concepto necesario para poder entender la distribución t Student. este concepto es "grados de libertad".

Para definir grados de libertad se hará referencia a la varianza muestral.

$$s^2 = \frac{\sum_{i=1}^{n} (x_i - \overline{x})^2}{n-1}$$

$$s^2 = \frac{\sum_{i=1}^{n} (x_i - \overline{x})^2}{n-1}$$

Esta fórmula está basada en n-1 grados de libertad.

Esta fórmula está basada en n-1 grados de libertad. Esta terminología

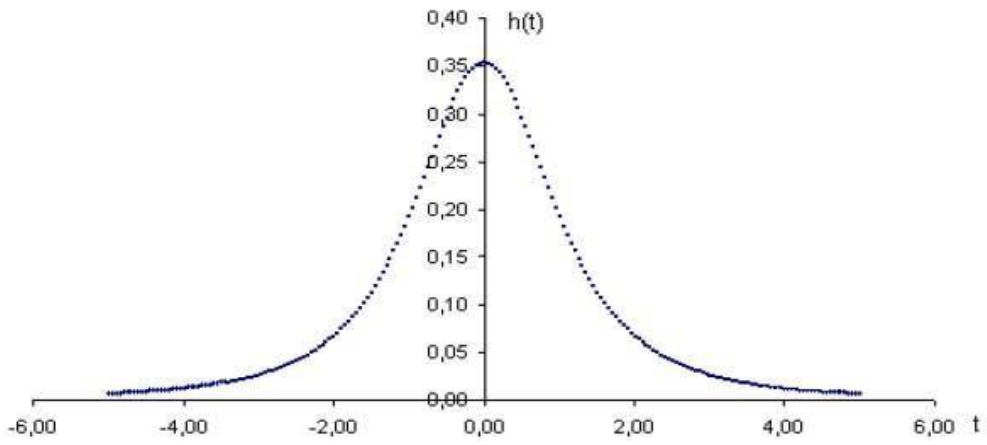
resulta del hecho de que si bien s2 está basada en n, éstas suman cero, así que especificar los valore de cualquier n-1 de las cantidades determina el valor restante.

Distribución de probabilidad t-Student

Una variable aleatoria se distribuye según el modelo de probabilidad t o T de Student con k grados de libertad, donde k es un entero positivo, si su función de densidades la siguiente:

$$h_k(t) = \frac{\Gamma\left(\frac{k+1}{2}\right)}{\Gamma\left(\frac{k}{2}\right)\sqrt{\pi k}}\left(1+\frac{t^2}{k}\right)^{-\frac{(k+1)}{2}} \quad -\infty < t < \infty, \quad donde \quad \Gamma(p) = \int_0^\infty e^{-x}x^{p-1}dx$$

La gráfica de esta función de densidad es simétrica, respecto del eje de ordenadas, con independencia del valor de k, y de forma algo semejante a la de una distribución normal:



La siguiente figura presenta la gráfica de varias distribuciones t. La apariencia general de la distribución t es similar a la de la distribución normal estándar: ambas son simétricas y unimodales, y el valor máximo de la ordenada se alcanza en la

media μ = 0. Sin embargo, la distribución t tiene colas más amplias que la normal; esto es, la probabilidad de las colas es mayor que en la distribución normal. A medida que el número de grados de libertad tiende a infinito, la forma límite de la distribución t es la distribución normal estándar.

2.2.3. ASOCIACIÓN Y CORRELACIÓN

El concepto de Asociación o Correlación Estadística entre dos variables es, en general, otro de los temas de Estadística Descriptiva que no está correctamente presentado en los textos básicos de Estadística. Por ello es frecuente encontrar falsas y absurdas conclusiones supuestamente basadas en el estudio de correlaciones estadísticas como ejemplos para desprestigiar los métodos estadísticos (ver “La Correlación no Implica Relación Causa Efecto”) Insistimos que esta situación sólo refleja la incultura estadística de quienes proponen tales conclusiones y que dicha incultura es de responsabilidad de los que estamos a cargo de la enseñanza de esta disciplina.

Una Conceptualización de Correlación Estadística

Una adecuada interpretación del concepto de asociación o correlación en Estadística es fundamental independizar su definición de cualquier sugerencia sobre la relación causa-efecto.

Para ello se debe precisar en qué sentido se dice que dos variables cuantitativas están asociadas o correlacionadas estadísticamente. Al respecto se sugiere utilizar la siguiente caracterización:

En Estadística Descriptiva se dice que dos variables cuantitativas “están asociadas”, “son dependientes”, o “están correlacionadas” si cuando se bien a aumentar (y se dice que la asociación o dependencia es directa o que la correlación es positiva)

ii) o bien a disminuir (y se dice que la asociación o dependencia es inversa o que la correlación es negativa)

Cuando no se presenta esta tendencia se dice que las variables no están asociadas o no son dependientes o no están correlacionadas.

Para completar la conceptualización de la Correlación Estadística es necesario destacar que la “tendencia” se refiere a la seguridad o grado de certeza respecto del hecho de que cuando una variable aumenta, la otra cumple o bien i) o bien ii). Si se

tiene una alta expectativa sobre esta ocurrencia, la Correlación es “alta”; si la expectativa de ocurrencia es “baja” la Correlación es baja.

No se debe confundir la calificación de “alto” o “bajo” con la magnitud del incremento o disminución. Los siguientes dibujos ilustran mejor esta situación.

2.2.4. ALGORITMOS.

Definición

“Un Algoritmo es una secuencia de operaciones detalladas y no ambiguas, que, al ejecutarse paso a paso, conducen a la solución de un problema”. En otras palabras es un conjunto de reglas para resolver una cierta clase de problema.

“Algoritmo es un conjunto de instrucciones que especifican la secuencia de operaciones a realizar, en orden, para resolver un sistema específico o clase de problema”.

“Un Algoritmo es la aplicación de pasos lógicos, secuenciales y metódicamente aplicados para dar solución a un problema en cuestión.” En otras palabras un algoritmo es una fórmula para resolver problemas.

Características De Los Algoritmos.

Las propiedades de un algoritmo son las siguientes:

a) El algoritmo debe ser preciso e indicar el orden de realización de cada paso.

b) El algoritmo debe ser definido, si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.

c) El algoritmo debe ser finito, si se sigue un algoritmo se debe terminar en algún momento; o sea debe tener un número finito de pasos.

El algoritmo debe ser planteado como un sistema de información.

Clasificación De Los Algoritmos

Se puede clasificar tomando en cuenta dos aspectos.

- Secuenciales
- Condicionales
- Repetitivos

Tipos De Algoritmos

Cualitativos: Son aquellos en los que se describen los pasos utilizando palabras.

Cuantitativos: Son aquellos en los que se utilizan cálculos numéricos para definir los pasos del proceso.

Lenguajes Algorítmicos

Es una serie de símbolos y reglas que se utilizan para describir de manera explícita un proceso.

Tipos De Lenguajes Algorítmicos

• Gráficos: Es la representación gráfica de las operaciones que realiza un algoritmo (diagrama de flujo).

• No Gráficos: Representa en forma descriptiva las operaciones que debe realizar un algoritmo (seudocódigo).

Un algoritmo puede ser expresado de las siguientes formas.

a) Lenguaje Natural: el uso de términos del lenguaje natural, es una forma de representar un algoritmo.

b) Lenguaje Simbólico: es otra forma de representación de un algoritmo, que además permite una introducción a la programación estructural.

c) Lenguaje Gráfico: es una forma de escribir una secuencia de pasos en forma de diagrama, en la práctica se denomina Diagramas de Flujo.

Una receta de un plato de cocina se puede expresar en español, inglés o francés pero cualquiera sea el lenguaje los pasos para la elaboración del plato se realizarán sin importar el cocinero.

Las Técnicas De Diseño De Algoritmos

El diseño de la mayoría de los algoritmos requiere creatividad y conocimientos de la metodología de programación, esto significa que debe desarrollar una lógica computacional a través de la resolución de programas.

Y principalmente en diseño de un algoritmo debe realizarse a partir de un análisis del problema.

El acto de diseñar un algoritmo puede considerarse como una tarea que difícilmente podrá ser del todo automatizada, todo problema algorítmico es un reto para su diseñador. Algunos resultan inmediatos de resolver, otros son bastante complejos. la investigación en esta área ha permitido descubrir un conjunto de métodos y esquemas de diseño hacia los cuales puede orientarse la realización de muchos algoritmos. Idear un algoritmo continúa siendo una labor bastante creativa

donde los conocimientos y las experiencias del propio diseñador tiene un papel fundamental.

Metodología Para La Solución De Problemas Por Medio De Computadora

La principal razón para que las personas aprendan lenguajes de programación es utilizar la computadora como una herramienta para la resolución de problemas. Dos fases pueden ser identificadas en el proceso de resolución de problemas.

• Fase de resolución del problema.

• Fase de implementación (realización) en un lenguaje de programación.

Definición del Problema

Esta fase está dada por el enunciado del problema, el cual requiere una definición clara y precisa.

Tomar en cuenta que la solución del problema debe estar en función de lo que el problema requiera y no en función de lo que el programador quiera.

Es importante que se conozca lo que se desea que realice la computadora; mientras esto no se conozca del todo no tiene mucho caso continuar con la siguiente etapa.

Análisis del Problema

No existe un método general para la resolución de problemas, la resolución de un problema es un proceso creativo donde el conocimiento, la habilidad y la experiencia tiene un papel importante, el proceder de manera sistemática ayuda a resolver un problema Al comenzar a abordar un problema hay que tener en cuenta que para la mayoría de ellos hay muchas maneras de resolver y pueden existir muchas soluciones, se plantean algunos criterios o estrategias generales que se deben tomar en cuenta, las cuales son útiles en el análisis del problema.

• Usar toda la información útil (no superficial) disponible en el enunciado del problema.

• Hacer explícita las reglas y datos que aparezcan implícitos (en muchos problemas numéricos se pueden utilizar reglas de la aritmética o álgebra)

• Profundizar en el problema considerado (emplear algún tipo de notación).

• Dividir el problema complejo en subproblemas más simples. que se pueden resolver independientemente y después combinar sus soluciones.

• Otra forma de abordar el problema consiste en trabajar “hacia atrás” es decir parir de la solución e intentar llegar al estado inicial.

El propósito del análisis del problema es ayudar al programador para llegar a una cierta comprensión de la naturaleza del problema. Una buena definición del problema junto con una descripción detallada de las especificaciones de entrada y de salida, son los requisitos más importantes para llegar a una solución eficaz.

Una vez que se ha comprendido lo que se desea de la computadora, es necesario definir:

• Los datos de entrada.

• Cual es la información que se desea producir (salida)

• Los métodos y fórmulas que se necesitan para procesar los datos.

Una recomendación muy practica es el que nos pongamos en el lugar de la computadora y analicemos que es lo que necesitamos que nos ordenen y en que secuencia para producir los resultados esperados.

Diseño del Algoritmo

Un computador no tiene capacidad para solucionar problemas más que cuando se le proporciona los sucesivos pasos a realizar. Estos pasos indican las instrucciones a ejecutar por la máquina y se denomina algoritmo.

Las características de un buen algoritmo son:

• Debe tener un punto particular de inicio.

• Debe ser definido, no debe permitir dobles interpretaciones.

• Debe ser general, es decir, soportar la mayoría de las variantes que se puedan presentar en la definición del problema.

• Debe ser finito en tamaño y tiempo de ejecución.

Se pueden utilizar cualquier técnica de diseño de algoritmos, diseño descendente, divide y vencerás.

Normalmente los pasos diseñados en un primer esbozo del algoritmo son incompletos e indican solo unos pocos pasos, tras esta primera descripción estos se amplían en una descripción más detallada con pasos específicos este proceso se denomina refinamiento del algoritmo.

Verificación del algoritmo

Una vez que se ha terminado de escribir un algoritmo es necesario comprobar que realiza las tareas para las que ha sido diseñado y produce el resultado correcto y esperado.

El modo más normal de comprobar un algoritmo es mediante la ejecución manual (prueba de escritorio), usando datos significativos que abarquen todo el posible rango de valores y anotando en una hoja de papel las modificaciones que se producen en las diferentes fases hasta la obtención de los resultados

Fase de Implementación

Una vez que el algoritmo está diseñado, representado gráficamente mediante una herramienta y verificado se debe pasar a la fase de codificación, traducir el algoritmo a un determinado lenguaje de programación que deberá ser completada con la ejecución y verificación de resultado en el computador

Codificación

La codificación es la operación de escribir la solución del problema (de acuerdo a la lógica del diagrama de flujo o pseudocodigo), en una serie de instrucciones detalladas, en un código reconocible por la computadora, la serie de instrucciones detalladas se le conoce como código fuente, el cual se escribe en un lenguaje de programación o lenguaje de alto nivel.

Prueba y Depuración

Los errores humanos dentro de la programación de computadoras son muchos y aumentan considerablemente con la complejidad del problema. El proceso de identificar y eliminar errores, para dar paso a una solución sin errores se le llama depuración.

La depuración o prueba resulta una tarea tan creativa como el mismo desarrollo de la solución, por ello se debe considerar con el mismo interés y entusiasmo.

Resulta conveniente observar los siguientes principios al realizar una depuración, ya que de este trabajo depende el éxito de nuestra solución.

Documentación

Es la guía o comunicación escrita es sus variadas formas, ya sea en enunciados, procedimientos, dibujos o diagramas.

A menudo un programa escrito por una persona, es usado por otra. Por ello la documentación sirve para ayudar a comprender o usar un programa o para facilitar futuras modificaciones (mantenimiento). La documentación se divide en tres partes:

- Documentación Interna
- Documentación Externa
- Manual del Usuario

Documentación Interna: Son los comentarios o mensaje que se añaden al código fuente para hacer más claro el entendimiento de un proceso.

Documentación Externa: Se define en un documento escrito los siguientes puntos:

- Descripción del Problema
- Nombre del Autor
- Algoritmo (diagrama de flujo o pseudocodigo)
- Diccionario de Datos
- Código Fuente (programa)

Manual del Usuario: Describe paso a paso la manera cómo funciona el programa, con el fin de que el usuario obtenga el resultado deseado.

Mantenimiento

Se lleva acabo después de terminado el programa, cuando se detecta que es necesario hacer algún cambio, ajuste o complementación al programa para que siga trabajando de manera correcta. Para poder realizar este trabajo se requiere que el programa este correctamente documentado.

Tecnicas Para La Formulación De Algoritmos

Para representar un algoritmo se debe utilizar algún método que permita independizar dicho algoritmo del lenguaje de programación elegido. Ello permitirá que un algoritmo pueda ser codificado inmediatamente en cualquier lenguaje.

Las herramientas utilizadas comúnmente para diseñar algoritmos son:

- Pseudocodigo
- Diagrama de Flujo.
- Diagramas Nassi/Shneiderman-S (Chapin)
- Método Warnier
- Método Jackson
- Método Bertini
- Método Tabourier

2.3. DEFINICIÓN DE TÉRMINOS BÁSICOS

- **Población o universo**. Conjunto de elementos que poseen una propiedad común, y que constituye la totalidad de individuos de interés para nuestro estudio.

- **Muestra.** Cualquier subconjunto de la población sobre el que se realizan los estudios para obtener conclusiones acerca de las características de la población.

- **Individuo.** Cada uno de los elementos de la muestra o de la población.

No tienen por qué ser objetos físicos.

- **Estadística descriptiva**. Recopilación y análisis de datos. Pueden ser datos referentes a toda la población o solo a una muestra, pero en este último caso no se pretende sacar conclusiones acerca de la población a la que pertenece la muestra.

- **Teoría de la probabilidad**. Es la teoría matemática en que se basa la posibilidad de realizar inferencias acerca de las propiedades de la población partiendo de la información contenida en la muestra.

- **Estadística inferencial o inductiva**. Utiliza la información que se desprende del análisis de una muestra para realizar una estimación de las propiedades de la población de la que se extrajo.

Modelamiento

Consiste en la habilidad para analizar un problema, resumir sus características esenciales, seleccionar y modificar las suposiciones básicas que caracterizan al sistema, y luego enriquecer y elaborar el modelo hasta obtener una aproximación útil.

Lenguaje De Programación

Vocabulario y conjunto de reglas gramaticales con el fin de indicarle a un computador como realizar tareas específicas. Cada lenguaje tiene un conjunto único de vocablos (palabras con semántica) y reglas de sintaxis.

Programa

Conjunto de instrucciones escritas en un lenguaje de programación que, traducidas a lenguaje de máquina, desempeñan una tarea determinada

Memoria

Son áreas de almacenamiento interno en el computador. Se puede pensar en la memoria como un arreglo de casillas, donde cada una puede almacenar información.

IDE

Entorno de Desarrollo Integrado, se usa este concepto para describir a un software que tiene un entorno de desarrollo donde se facilita la programación visual con el mouse liberando al programador del tedioso trabajo del diseño de interfaces desde cero.

Algoritmo

Conjunto de pasos ordenados para resolver un problema, como una fórmula matemática o las instrucciones de un programa.

Analista

Persona que realiza el análisis de un problema. Tras el análisis del problema el programador puede confeccionar el programa para computadora a partir de las indicaciones del análisis.

2.4. HIPOTESIS

2.4.1. Hipótesis General

Los algoritmos de código abierto mejorarán en el tratamiento de datos usando las técnicas y métodos estadísticos de las diferentes asignaturas en la escuela profesional de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonia Peruana.

2.4.2. Variables

VARIABLE INDEPENDIENTE (X) : TECNICAS ESTADISTICAS
VARIABLE DEPENDIENTE (Y) : ALGORITMOS

Variables	Dimensiones	Indicadores
INDEPENDIENTE (X) TECNICAS ESTADISTICAS	X1. ESTADISTICA DESCRIPTIVA X2. ESTADISTICA COMPARATIVA	X11. Medidas de tendencia central X12. Medidas de dispersión X21. T de Studen
DEPENDIENTE (Y) **ALGORITMOS**	Y1. ALGORITMOS ESTADISTICOS	Y11. Programación de algoritmos

CAPITULO III: METODOLOGIA

3.1. TIPO DE INVESTIGACION

El presente estudio por el grado de cuantificación reúne las condiciones de una investigación tipo básico y por sus características es de nivel descriptivo.

3.2. DISEÑO DE LA INVESTIGACION

El diseño de la investigación es el no experimental de corte transversal o transeccional.

Los estudios no experimentales son aquellos donde no hay manipulación den las variables en su análisis, Gómez (2013) manifiesta que los diseños no experimentales trabajan con muestras grandes, utilizan variables independiente y dependiente, pueden ser de dos clases: transeccionales y longitudinales y no se manipulan las variables. (p.110)

3.3. POBLACION

La unidad de análisis del estudio corresponde a los estudiantes de la Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonia Peruana

3.4. MUESTRA

La muestra para la presente investigación estuvo constituida por 60 estudiantes de la Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonia Peruana, la selección de la muestra se efectuó en forma no probabilística intencionada.

3.5. TECNICAS DE INVESTIGACION

3.5.1. Técnica de recolección de datos

Investigación documental. La hoja de codificación

Es el instrumento del análisis de contenido que consiste en una hoja impresa que contiene las unidades de análisis, las categorías y subcategorías.

3.5.2. Instrumento de recolección de datos

Análisis de contenido- codificación, se define como la técnica de investigación para la descripción objetiva, sistemática y cuantitativa del contenido manifiesto de la comunicación

El análisis de contenido se realiza mediante tres tareas principales:

- Establecer las unidades de análisis
- Determinar las categorías de análisis
- Seleccionar una muestra del material de análisis. (Ander-Egg,1974:199)

3.6. Procesamiento de la informacion

Análisis de datos

El análisis de datos significa como van a ser tratados los datos recolectados en sus diferentes etapas. Para ello se emplea la estadística como una herramienta. “Esta fase se presenta posterior a la aplicación del instrumento y finalizada la recolección de los datos. Se procederá a aplicar el análisis de los datos para dar respuesta a las interrogantes de la investigación” (Hevia, 2001, p. 46).

CAPITULO IV: RESULTADOS Y DISCUSION

ALGORITMOS ESTADISTICOS

Chi Cuadrado

```
ChiCuadrado var
factor : doble;   // factor con multiplies sumas de series
g     : doble;   // lngamma(k1+1)
k1     : doble;   // adjuste con grados de libertad
sum    : doble;   // Almacenamiento parcial de sumas term   : doble;   // termino de series
x1     : doble;   // argumento de función ajustada chi2prob : doble; // Probabilidad chi-cuadrada
inicio
si (X < 0.01) or (X > 1000.0) entonces inicio
si X < 0.01 entonces chi2prob := 0.0001 sino chi2prob := 0.999;
fin
sino inicio
x1 := 0.5 * X;
k1 := 0.5 * k;
g := gammln(k1 + 1);
factor := exp(k1 * ln(x1) – g – x1);
sum := 0.0;
si factor > 0 entonces inicio
term := 1.0;
sum := 1.0;
mientras ((term / sum) > 0.000001) do inicio
k1 := k1 + 1;
term  := term * (x1 / k1);
sum := sum + term;
fin;
chi2prob := sum * factor;
fin; //fin si .. sino
Result := chi2prob;
Fin;
Fin;
```

4.1.2. Algoritmo de Matriz inversa

```
matinv (VAR a, vtimesw, v, w: hacerbleMat; n: entero);
//Descomposicion de matrices
LABEL 1,2,3;
VAR
ainversa : array de array de hacerble; m,mp,np,nm,l,k,j,its,i: entero; z,y,x,scalar,s,h,g,f,c,anorm: hacerble;
rv1: array de hacerble;

inicio setlength(rv1,n); setlength(ainversa,n,n); m := n;
mp := n; np := n; g := 0.0;
scalar := 0.0;
anorm := 0.0;
desde i := 0 hasta n-1 hacer inicio
l := i+1;
rv1[i] := scalar*g;
g := 0.0;
s := 0.0;
scalar := 0.0;
si (i <= m-1) entonces inicio
desde k := i hasta m-1 hacer inicio
scalar := scalar+abs(a[k,i])
fin;
si (scalar <> 0.0) entonces inicio
desde k := i hasta m-1 hacer inicio
a[k,i] := a[k,i]/scalar;
s := s+a[k,i]*a[k,i]
fin;
f := a[I,i];
g := -sign(sqrt(s),f);
h := f*g-s;
a[I,i] := f-g;
si (I <> n-1) entonces inicio
desde j := l hasta n-1 hacer inicio
s := 0.0;
desde k := i hasta m-1 hacer inicio
s := s+a[k,i]*a[k,j]
fin;
f := s/h;
desde k := i hasta m-1 hacer inicio
a[k,j] := a[k,j]+
f*a[k,i]
fin fin
fin;
desde k := i hasta m-1 hacer inicio
a[k,i] := scalar*a[k,i]
fin fin
fin;
w[I,i] := scalar*g;
g := 0.0; s := 0.0; scalar := 0.0;
si ((i <= m-1) y (i <> n-1)) entonces inicio
desde k := l hasta n-1 hacer inicio
scalar := scalar+abs(a[i,k])
fin;
si (scalar <> 0.0) entonces inicio
desde k := l hasta n-1 hacer inicio
a[I,k] := a[I,k]/scalar;
s := s+a[I,k]*a[I,k]
fin;

f := a[I,l];
```

```
g := -sign(sqrt(s),f);
h := f*g-s;
a[i,l] := f-g;
desde k := l hasta n-1 hacer inicio
rv1[k] := a[I,k]/h fin;
si (i <> m-1) entonces inicio
desde j := l hasta m-1 hacer inicio
s := 0.0;
desde k := l hasta n-1 hacer inicio
s := s+a[j,k]*a[I,k]
fin;
desde k := l hasta n-1 hacer inicio
a[j,k] := a[j,k]
+s*rv1[k]
fin fin
fin;
desde k := l hasta n-1 hacer inicio
a[i,k] := scalar*a[i,k]
fin fin
fin;
anorm := max(anorm,(abs(w[I,i])+abs(rv1[i])))
fin;
desde i := n-1 hacerWNhasta 0 hacer inicio
si (i < n-1) entonces inicio
si (g <> 0.0) entonces
inicio
desde j := l hasta n-1 hacer inicio
v[j,i] := (a[I,j]/a[I,l])/g fin;
desde j := l hasta n-1 hacer
inicio
s := 0.0;
desde k := l hasta n-1 hacer inicio
s := s+a[I,k]*v[k,j]
fin;
desde k := l hasta n-1 hacer inicio
v[k,j] := v[k,j]+s*v[k,i]
fin fin
fin;
desde j := l hasta n-1 hacer inicio
v[I,j] := 0.0;
v[j,i] := 0.0
fin fin;
v[I,i] := 1.0;
g := rv1[i];
l := i fin;
desde i := n-1 hacerWNhasta 0 hacer inicio
l := i+1;
g := w[I,i];
si (i < n-1) entonces inicio
desde j := l hasta n-1 hacer inicio
a[i,j] := 0.0 fin
fin;
si (g <> 0.0) entonces inicio
g := 1.0/g;
si (i <> n-1) entonces inicio
desde j := l hasta n-1 hacer inicio
s := 0.0;
desde k := l hasta m-1 hacer inicio
s := s+a[k,i]*a[k,j]
fin;
f := (s/a[i,i])*g;
desde k := i hasta m-1 hacer inicio
a[k,j] := a[k,j]+f*a[k,i]
```

```
fin fin
fin;
desde j := i hasta m-1 hacer inicio
a[j,i] := a[j,i]*g
fin
fin sino inicio
desde j := i hasta m-1 hacer inicio
a[j,i] := 0.0 fin
fin;
a[i,i] := a[i,i]+1.0 fin;
desde k := n-1 hacerWNhasta 0 hacer
inicio
desde its := 1 hasta 30 hacer inicio
desde l := k hacerWNhasta 0 hacer inicio
nm := l-1;
si ((abs(rv1[l])+anorm) = anorm) entonces GOTO 2;
si ((abs(w[nm,nm])+anorm) = anorm) entonces GOTO 1 fin;
1:
s := 1.0;
desde i := l hasta k hacer inicio
f := s*rv1[i];
si ((abs(f)+anorm) <> anorm) entonces inicio
g := w[I,i];
h := sqrt(f*f+g*g);
w[I,i] := h; h := 1.0/h; c := (g*h); s := -(f*h);
desde j := 0 hasta m-1 hacer inicio
y := a[j,nm];
z := a[j,i];
a[j,nm] := (y*c)+(z*s);
a[j,i] := -(y*s)+(z*c)
fin fin
fin;
2:      z := w[k,k];
si (l = k) entonces inicio
si (z < 0.0) entonces
inicio w[k,k] := -z;
desde j := 0 hasta n-1 hacer
inicio v[j,k] := -v[j,k]
fin fin; GOTO 3 fin;
si (its = 30) entonces
inicio
showmessage('No converge en 30 iteraciones');
exit;
fin;
x := w[l,l];
nm := k-1;
y := w[nm,nm];
g := rv1[nm];
h := rv1[k];
f := ((y-z)*(y+z)+(g-h)*(g+h))/(2.0*h*y);
g := sqrt(f*f+1.0);
f := ((x-z)*(x+z)+h*((y/(f+sign(g,f)))-h))/x;
c := 1.0;
s := 1.0;
desde j := l hasta nm hacer inicio
I := j+1;
g := rv1[i]; y := w[I,i]; h := s*g;
g := c*g;
z := sqrt(f*f+h*h);
rv1[j] := z;
c := f/z;
s := h/z;
```

```
f := (x*c)+(g*s);
g := -(x*s)+(g*c);
h := y*s;
y := y*c;
desde nm := 0 hasta n-1 hacer inicio
x := v[nm,j];
z := v[nm,i];
v[nm,j] := (x*c)+(z*s);
v[nm,i] := -(x*s)+(z*c)
fin;
z := sqrt(f*f+h*h);
w[j,j] := z;
si (z <> 0.0) entonces inicio
z := 1.0/z;
c := f*z;
s := h*z fin;
f := (c*g)+(s*y);
x := -(s*g)+(c*y);
desde nm := 0 hasta m-1 hacer inicio
y := a[nm,j];
z := a[nm,i];
a[nm,j] := (y*c)+(z*s);
a[nm,i] := -(y*s)+(z*c)
fin fin;
rv1[l] := 0.0; rv1[k] := f; w[k,k] := x
fin;
3:   fin;
desde i := 0 hasta n-1 hacer desde j := 0 hasta n-1 hacer inicio
si w[j,j] < 1.0e-6 entonces vtimesw[i,j] := 0 sino vtimesw[I,j] := v[I,j] * (1.0 / w[j,j] );
fin;
desde i := 0 hasta m-1 hacer desde j := 0 hasta n-1 hacer inicio
ainversa[i,j] := 0.0;
desde k := 0 hasta m-1 hacer inicio
ainversa[I,j] := ainversa[I,j] + vtimesw[I,k] * a[j,k]
fin;
fin;
desde i := 0 hasta n-1 hacer desde j := 0 hasta n-1 hacer
a[i,j] := ainversa[i,j];
ainversa := nil;
rv1 := nil;
fin;
```

Algoritmo de Chi Inversa

```
inversachi(p : doble; k : ientero)
var
a1, w, z : double;
begin
z := inversez(p);
a1 := 2.0 / ( 9.0 * k);
w := 1.0 - a1 + z * sqrt(a1); Result := (k * w * w * w);
end;
```

Algoritmo de T de Student

```
funcion STUDENT(q,v,r : real) : real;

{ Rinde la probabilidad de un valor de la muestra de Q o mayo de una población con medios y r grados de libertad para el
error medio cuadrado de v.}
var
probq : real;
ifault : entero;
funcion alnorm(x: real; upper: boleano): real;
var
ltone, utzero, zero, half, one, con, z, y : real;
up : boleano;
altemp: real;
inicio
ltone := 7.0; utzero := 18.66; zero := 0.0;
half := 0.5; one := 1.0; con := 1.28; up := upper; z := x;
si z < zero entonces inicio
up := not up;
z := -z;
fin
si (z <= ltone) o (up) y (z <= utzero) entonces inicio
y := half * z * z;
si z > con entonces inicio
altemp := 0.398942280385 * exp(-y) / (z - 3.8052e-8 + 1.00000615302 /
(z + 3.98064794e-4 + 1.98615381364 / (z - 0.151679116635 + 5.29330324926 /
(z + 4.8385912808 - 15.1508972451 /
(z + 0.742380924027 + 30.789933034 / (z + 3.99019417011))))));
fin
sino altemp := half - z * (0.398942280444 - 0.399903438504 * y / (y + 5.75885480458 - 29.8213557808 /
(y + 2.62433121679 + 48.6959930692 / (y + 5.92885724438))));
fin
sino altemp := zero;
si not up entonces altemp := one - altemp;
alnorm := altemp;
fin
```

Algoritmo de ordenada

```
funcion ordinate (z: double): double;
var pi : double;
inicio
pi := 3.14159;
Result := (1.0 / sqrt(2.0 * pi)) * (1.0 / exp(z * z / 2.0));
fin // Fin de funcion
//-----------------------------------------------------------------

procedure Rank(v1col : entero; VAR Values : DblDyneVec);
var
pcntiles, CatValues : DblDyneVec;
freq : IntDyneVec;
i, j, nocats : entero;
Temp, cumfreq, upper, lower : double;

inicio
SetLength(freq, NoCases); SetLength(pcntiles, NoCases); SetLength(CatValues, NoCases);

// obtener valores que se clasifican en los valores del vector fo i := 1 to NoCases do
Values[i-1] := StrToFloat(PrincipalFrm.DataGrid.Cells[v1col,i]);

// ordenar los valores
fo i := 1 to NoCases - 1 do //ordenar del mas alto al mas bejo inicio
fo j := i + 1 to NoCases do inicio
si (Values[i-1] < Values[j-1]) entonces // swap inicio
Temp := Values[i-1]; Values[i-1] := Values[j-1];
Values[j-1] := Temp;
fin fin
fin
// ahora obtener ninguna. valores de únicos y frecuencia nocats := 1;
fo i := 1 to NoCases do freq[i-1] := 0; Temp := Values[0];
CatValues[0] := Temp; fo i := 1 to NoCases do inicio
si (Temp = Values[i-1]) entonces  freq[nocats-1] := freq[nocats-1] + 1
sino // nuevo valores inicio
nocats := nocats + 1;
freq[nocats-1] := freq[nocats-1] + 1;
Temp := Values[i-1]; CatValues[nocats-1] := Temp;
fin
fin
// obtener rangos cumfreq := 0.0;
fo i := 1 to nocats do inicio
upper := NoCases-cumfreq;
cumfreq := cumfreq + freq[i-1];
lower := NoCases - cumfreq + 1;
pcntiles[i-1] := (upper - lower) / 2.0 + lower;
fin
// convertir valores originales correspondiente al rango fo i := 1 to NoCases do
inicio
Temp := StrToFloat(PrincipalFrm.DataGrid.Cells[v1col,i]);
fo j := 1 to nocats do inicio
si (Temp = CatValues[j-1]) entonces Values[i-1] := pcntiles[j-1];
fin fin
// limpiar
CatValues := nil; pcntiles := nil; freq := nil;
Fin
```

4.1.8. Algoritmo punto de rango

```
procedure PRank(v1col : entero; VAR Values : DblDyneVec);
// calcula los rangos percentiles de los valores almacenados en la tabla de
// datos en la columna v1col var
pcntiles, cumfm, CatValues : DblDyneVec;
freq, cumf : IntDyneVec; Temp : double;
i, j, nocats : entero;

inicio
SetLength(freq, NoCases); SetLength(pcntiles, NoCases); SetLength(cumf, NoCases);
SetLength(cumfm, NoCases);
SetLength(CatValues, NoCases);

// obtener valores que se clasifican en los valores del vector fo i := 1 to NoCases do
Values[i-1] := StrToFloat(PrincipalFrm.DataGrid.Cells[v1col,i]);

// ordenar los valores
fo i := 1 to NoCases - 1 do //orden de mayo a menor inicio
fo j := i + 1 to NoCases do
inicio

si (Values[i-1] < Values[j-1]) entonces // swap inicio
Temp := Values[i-1];
Values[i-1] := Values[j-1]; Values[j-1] := Temp;
fin
fin fin
nocats := 1;
fo i := 1 to NoCases do freq[i-1] := 0; Temp := Values[0];
CatValues[0] := Temp;
fo i := 1 to NoCases do inicio
si (Temp = Values[i-1])entonces freq[nocats-1] := freq[nocats-1] + 1 sino // nuevo valo
inicio
nocats := nocats + 1;
freq[nocats-1] := freq[nocats-1] + 1; Temp := Values[i-1]; CatValues[nocats-1] := Temp;
fin
fin
// encontrar nueva frecuencia acumulada cumf[nocats-1] := freq[nocats-1];
fo i := nocats - 2 downto 1 do cumf[i-1] := freq[i-1] + cumf[i];
// conseguir frecuencia acumulada de puntos medios y rangos percentiles cumfm[nocats-1] := freq[nocats-1];
pcntiles[nocats-1] := (cumf[nocats-1] / 2.0) / NoCases;
fo i := nocats - 2 downto 1 do inicio
cumfm[i-1] := (freq[i-1] / 2.0) + cumf[i];
pcntiles[i-1] := cumfm[i-1] / NoCases;
fin
// convertir los valores originales a sus rangos percentiles correspondientes fo i := 1 to NoCases do
inicio
Temp := StrToFloat(PrincipalFrm.DataGrid.Cells[v1col,i]);
fo j := 1 to nocats do inicio
si (Temp = CatValues[j-1]) entonces Values[i-1] := pcntiles[j-1];
fin fin
//limpiar el montón CatValues := nil; cumfm := nil;
cumf := nil; pcntiles := nil; freq := nil;
fin
```

Algoritmo de estadística básica

```
funcion UniStats(N : entero; VAR X : DblDyneVec; VAR z : DblDyneVec;

VAR Mean : double; VAR varianza : double; VAR SD : double;
VAR Skew : double; VAR Kurtosis : double; VAR SEmean : double; VAR SESkew : double; VAR SEkurtosis : double; VAR min :
double; VAR max : double; VAR Range : double; VAR MissValue : string) : entero;
VAR
NoGood : entero; // Nº de buenos casos devuelto po la función i : entero; // Índice de bucles
num, den, cases, sum, M2, M3, M4, deviation, devsqr : double;
valuestr : string;

                                                                    inicio
Mean := 0.0; varianza := 0.0; SD := 0.0;
Skew := 0.0;
Kurtosis := 0.0; SEmean := 0.0; SESkew := 0.0; SEKurtosis := 0.0; min := 1.0e20; max := -1.0e20; range := 0.0; NoGood := 0;
sum := 0.0; M2 := 0.0; M3 := 0.0; M4 := 0.0;

fo i := 0 to N-1 do inicio
ValueStr := FloatToStr(X[i]);
si Trim(MissValue) = ValueStr entonces continue; NoGood := NoGood + 1;
sum := sum + X[i];
varianza := varianza + (X[i] * X[i]); si X[i] < min entonces min := X[i]; si X[i] > max entonces max := X[i];
fin

si NoGood > 0 entonces inicio
Mean := sum / NoGood;
range := max - min;
fin

si NoGood > 1 entonces inicio
varianza := varianza - (sum * sum) / NoGood;
varianza := varianza / (NoGood - 1); SD := sqrt(varianza);
SEmean := sqrt(varianza / NoGood);
fo i := 0 to N-1 do inicio
ValueStr := FloatToStr(X[i]);
si Trim(MissValue) = ValueStr entonces continue;
deviation := X[i] - Mean;
z[i] := deviation / SD;

devsqr := deviation * deviation; M2 := M2 + devsqr;
M3 := M3 + (deviation * devsqr);
M4 := M4 + (devsqr * devsqr);
fin fin
si NoGood > 3 entonces inicio
Skew := (NoGood * M3) / ((NoGood - 1) * (NoGood - 2) * SD * varianza);
num := 6.0 * NoGood * (NoGood - 1);
den := (NoGood - 2) * (NoGood + 1) * (NoGood + 3); SESkew := sqrt(num / den);
Kurtosis := (NoGood * (NoGood + 1) * M4) - (3.0 * M2 * M2 * (NoGood - 1)); Kurtosis := Kurtosis / ((NoGood - 1) * (NoGood
- 2) * (NoGood - 3) *
                                                                    (varianza * varianza));
SeKurtosis := sqrt((4.0 * (NoGood * NoGood - 1) * (SESkew * SESkew)) /
((NoGood - 3) * (NoGood + 5)));
fin
Result := NoGood;
fin
//------------------------------------------------------------------ fin.
```

Algoritmo que calcula los rangos percentiles

```
funcion PRank(v1col : entero; VAR Values : DblDyneVec);
// calcula los rangos percentiles de los valores almacenados en la tabla de
// datos en la columna v1col var
pcntiles, cumfm, CatValues : DblDyneVec;
freq, cumf : IntDyneVec; Temp : haceruble;
i, j, nocats : entero;
inicio
SetLength(freq, Nocasos); SetLength(pcntiles, Nocasos); SetLength(cumf, Nocasos); SetLength(cumfm, Nocasos);
SetLength(CatValues, Nocasos);
// obtener valores que se clasifican en los valores del vector para i := 1 a Nocasos hacer
                                        Values[i-1] := StrToFloat(PrincipalFrm.DataGrid.Cells[v1col,i]);
// ordenar los valores
para i := 1 a Nocasos - 1 hacer //orden de mayo a menor
inicio
para j := i + 1 a Nocasos hacer inicio
si (Values[i-1] < Values[j-1]) entonces // swap inicio
Temp := Values[i-1]; Values[i-1] := Values[j-1];
Values[j-1] := Temp;
fin fin
fin
nocats := 1;
para i := 1 a Nocasos hacer freq[i-1] := 0; Temp := Values[0];
CatValues[0] := Temp;
para i := 1 a Nocasos hacer inicio
si (Temp = Values[i-1])entonces freq[nocats-1] := freq[nocats-1] + 1 sino // nuevo valor
inicio
nocats := nocats + 1;
freq[nocats-1] := freq[nocats-1] + 1; Temp := Values[i-1]; CatValues[nocats-1] := Temp;
fin fin
// encontrar nueva frecuencia acumulada
cumf[nocats-1] := freq[nocats-1];
para i := nocats - 2 hacerwnto 1 hacer cumf[i-1] := freq[i-1] + cumf[i];
// conseguir frecuencia acumulada de puntos medios y rangos percentiles cumfm[nocats-1] := freq[nocats-1];
pcntiles[nocats-1] := (cumf[nocats-1] / 2.0) / Nocasos;
para i := nocats - 2 hacerwnto 1 hacer inicio
cumfm[i-1] := (freq[i-1] / 2.0) + cumf[i];
pcntiles[i-1] := cumfm[i-1] / Nocasos;
fin
// convertir los valores originales a sus rangos percentiles correspondientes para i := 1 a Nocasos hacer
inicio
Temp := StrToFloat(PrincipalFrm.DataGrid.Cells[v1col,i]);
para j := 1 a nocats hacer inicio
si (Temp = CatValues[j-1]) entonces Values[i-1] := pcntiles[j-1];
fin fin
//limpiar el montón
CatValues := nil; cumfm := nil; cumf := nil; pcntiles := nil; freq := nil;
fin
```

CAPITULO V: CONCLUSIONES

5.1. Conclusiones Específicas

- Los algoritmos estadísticos cumplen la función de instruir a los usuarios en el manejo de procesos estadísticos, de los cuales el estudiante podrá tener una mejor idea de los procedimientos matemáticos necesarios para resolver los distintos postulados estadísticos.

- Se ha desarrollado algoritmos computaciones que procesan postulados de estadística básica en un lenguaje de programación.

- Se ha integrado los algoritmos computacionales en un entorno de procesamiento de datos de forma similar a un software estadístico comercial.

5.2. Conclusión General

Se desarrollo algoritmos computacionales con funciones básicas para ser analizado y usados por estudiantes de la Facultad de ingeniería de sistemas e informática de la universidad Nacional de la Amazonia Peruana.

CAPITULO VI: RECOMENDACIONES

- Se sugiere incluir el dictado de cursos de algoritmia estadística en los cursos de programación, así como incluir programación en los cursos de estadística.

- Se recomienda hacer especial énfasis en la programación orientada a objetos y talleres de estadística computacional.

.

CAPITULO VII: REFERENCIAS BIBLIOGRAFICAS

David C Lay, Jesús Murrieta Murrieta (2007), Álgebra lineal y sus aplicaciones, (Tercera Edición) Pearson Educación

David Moore (2006), Estadística Aplicada Básica, Antoni Bosch editor.

Dirk Louis, José Luis Cortés (2000), Delphi 5, Alfaomega Marcombo.

Francisco Charte Ojeda (2003), DELPHI 7 GUIA PRÁCTICA, Anaya Multimedia.

Gary Cornell (1996), Delphi para programadores, McGraw-Hill/Interamericana de Espana, S.A.

Ian Marteens (2003), La cara oculta de Delphi 6, Danysoft Internacional.

Ibañez Quispe Vladimiro (1998), Modelos Lineales, Universidad Nacional del Altiplano – Facultad de Ingeniería Estadística e Informática.

J.M. Juran, R.S. Bingham, Frank M. Gryna, José María Vallhonrat (1983), Manual de control de la calidad, Editorial Reverté.

Jaime Serret Moreno-Gil (1995), Manual de Estadística Universitaria, ESIC Editorial.

Jeff Sutherland (2007), Pretty Good Scrum: Secret Sauce For Distributed Teams,

Scrum Trained Institute.

Julián de la Horra Navarro (2003), Estadística Aplicada, Ediciones Díaz de Santos.

Ludovic Dubois (2000), Delphi: Desarrollo rápido de aplicaciones bajo Windows,

Gestión 2000.

Palomino Quispe, Platón (1996), Diseños y Técnicas de Investigación (1er edición), Titicaca FCEDUR-UNA-Puno.

Naresh K. Malhotra, José Francisco Javier Dávila Martínez (2004), Investigación de Mercados, Pearson Educación.

Pressman Roger S. (1998) Ingeniería del software: Un enfoque práctico. Mc

Graw Hill, 4ta. Edición.

Sharon L. Lohr, scar Alfredo (trad.) Palmas Velasco, Carlos (rev.) Martínez

Reyes (2000), Muestreo: Diseño y análisis, International Thomson.

ANEXO 3: MATRIZ DE CONSISTENCIA

PROBLEMAS	OBJETIVOS	HIPOTESIS	VARIABLE	DIMENSION	INDICADORES
¿De qué manera es posible desarrollar la integración de base teórica de la estadística en algoritmos computacionales?	**OBJETIVO GENERAL** Desarrollar algoritmos computacionales para los métodos y técnicas estadísticas más usados para estudiantes de la Facultad de ingeniería de sistemas e informática de la universidad Nacional de la Amazonia Peruana. **OBJETIVOS ESPECIFICOS**) Diseñar algoritmos computacionales para los métodos y técnicas estadísticas más usados.) Construir módulos de código abierto de los algoritmos generados en un lenguaje de programación.) Integrar diferentes módulos con características de fácil uso y programación amigable.	Los algoritmos de código abierto mejorarán en el tratamiento de datos usando las técnicas y métodos estadísticos de las diferentes asignaturas en la escuela profesional de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonia Peruana.	VARIABLE INDEPENDIENTE (X) : TECNICAS ESTADISTICAS VARIABLE DEPENDIENTE (Y) : ALGORITMOS	X1. ESTADISTICA DESCRIPTIVA X2. ESTADISTICA COMPARATIVA Y1. ALGORITMOS ESTADISTICOS	X11. Medidas de tendencia central X12. Medidas de dispersión X21. T de Studen Y11. Programación de algoritmos

Algoritmos para el calculo de funciones estadisticas

Maycol Alcalde Moncada
FISI-UNAP[1]
Iquitos, Perú

Abstract— Research as your objective To develop computer algorithms for the most used statistical methods and techniques for students of the School of Systems Engineering and Computer Science of the National University of the Peruvian Amazon.
, the research belongs to the basic type with a descriptive level and the design was the non-experimental transversal type, whose results are presented by the statistical algorithms. The sample for the present investigation consisted of 60 students from the Faculty of Systems Engineering and Computer Science. The National University of the Peruvian Amazon, the selection of the sample was carried out in an intentional non-probabilistic manner.
.
The technique used in the data collection was documentary research. The coding sheet and the instrument was content analysis, concluding: The statistical algorithms fulfill the function of instructing the users in the handling of statistical processes, of which the student will have a better idea of the mathematical procedures necessary to solve the different statistical postulates.

INTRODUCCION

En la Escuela Profesional de Ingeniería de sistemas e Informática de la Universidad Nacional de la Amazonia Peruana es habitual que un estudiante se vea limitado en su quehacer académico por los programas como SPSS, Minitab, SAS, NCSS, etc. Estas herramientas se usan a nivel de usuario cuando la necesidad de muchos estudiantes es entender algoritmos computacionales con código fuente visible aplicados a los cursos de estadística. En las asignaturas de estadísticas se estudian en su mayoría la teoría, conceptos, teoremas, funciones estadísticas, métodos y técnicas, cuya aplicación se hace mediante el uso de cálculos científicos de dicho análisis estadístico, en las asignaturas de informática se da una formación netamente centrada en el área, creando así un divorcio involuntario entre ambas asignaturas, siendo necesario herramientas de software libre que facilite el estudio coordinado integrando.

En la actualidad las asignaturas de estadística se imparte base teórica, pero no se desarrollan algoritmos computacionales de su contenido en el área de informática ocurre de manera similar, se dictan cursos taller de programación centrada en algoritmos genéricos omitiendo algoritmos de estadística aplicada, cuando en la actualidad la estadística constituye técnicas y métodos de optimización, complejidad y eficiencia de los algoritmos computacionales, los cuales deben ser implementados en un software aplicado en las asignaturas para comprender en las técnicas y métodos, es importante generar algoritmos computacionales y más adelante sea implementado el código fuente como parte del taller de las asignaturas de informática.

Para el estudio de algoritmos estadísticos existe en Internet código fuente limitado, superficial y disperso, que no permite ser analizado adecuadamente variando en metodología, lenguaje en el que fue escrito y la documentación de diverso tipo; por otro lado tenemos la existencia de código fuente de calidad en programas avanzados como R ó PSPP, cuyos comentarios no están en nuestro idioma, sus algoritmos son muy refinados siendo orientados para su aplicación en ambientes científicos de primer nivel lo que dificulta su entendimiento para un estudiante de pregrado, es decir están al alcance del entendimiento de muy pocos estudiantes de la escuela profesional con avanzado conocimiento de programación.

Considerando estos factores es necesario generar algoritmos computacionales sencillos y eficientes para cada una de las técnicas y métodos estadísticos estudiados en las aulas, de manera que estas sean entendidas y se puedan implementar en programas.

cuentan con material que cubra el contenido curricular básico, según datos del MINEDU al menos 26 lenguas cuentan con alfabeto oficial. Así mismo existe reducida incorporación de contenido cultural de cada pueblo.

[1] Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonía Peruana.

METODOLOGÍA

El presente estudio por el grado de cuantificación reúne las condiciones de una investigación tipo básico y por sus características es de nivel descriptivo.

El diseño de la investigación es el no experimental de corte transversal o transeccional.
Los estudios no experimentales son aquellos donde no hay manipulación den las variables en su análisis, Gómez (2013) manifiesta que los diseños no experimentales trabajan con muestras grandes, utilizan variables independiente y dependiente, pueden ser de dos clases: transeccionales y longitudinales y no se manipulan las variables. (p.110)
La unidad de análisis del estudio corresponde a los estudiantes de la Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonia Peruana
La muestra para la presente investigación estuvo constituida por 60 estudiantes de la Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional de la Amazonia Peruana, la selección de la muestra se efectuó en forma no probabilística intencionada.

Técnica de recolección de datos

Investigación documental. La hoja de codificación

Es el instrumento del análisis de contenido que consiste en una hoja impresa que contiene las unidades de análisis, las categorías y subcategorías.

Instrumento de recolección de datos

Análisis de contenido- codificación, se define como la técnica de investigación para la descripción objetiva, sistemática y cuantitativa del contenido manifiesto de la comunicación

III.Resultados

Algoritmo de estadística básica

```
funcion UniStats(N : entero; VAR X : DblDyneVec; VAR z : DblDyneVec;

VAR Mean : double; VAR varianza : double; VAR SD : double;
VAR Skew : double; VAR Kurtosis : double; VAR SEmean : double; VAR SESkew : double; VAR SEkurtosis : double; VAR min : double; VAR max : double; VAR Range : double; VAR MissValue : string) : entero;
VAR
NoGood : entero; // Nº de buenos casos devuelto po la función i : entero; // Índice de bucles
```

Fig. 1: Boceto de la distribución del contenido de la aplicación

```
num, den, cases, sum, M2, M3, M4, deviation, devsqr : double;
valuestr : string;
inicio
Mean := 0.0; varianza := 0.0; SD := 0.0;
Skew := 0.0;
Kurtosis := 0.0; SEmean := 0.0; SESkew := 0.0; SEKurtosis := 0.0; min := 1.0e20; max := -1.0e20; range := 0.0; NoGood := 0;
sum := 0.0; M2 := 0.0; M3 := 0.0; M4 := 0.0;

fo i := 0 to N-1 do inicio
ValueStr := FloatToStr(X[i]);
si Trim(MissValue) = ValueStr entonces continue; NoGood := NoGood + 1;
sum := sum + X[i];
varianza := varianza + (X[i] * X[i]); si X[i] < min entonces min := X[i]; si X[i] > max entonces max := X[i];
fin

si NoGood > 0 entonces inicio
Mean := sum / NoGood;
range := max - min;
fin

si NoGood > 1 entonces inicio
varianza := varianza - (sum * sum) / NoGood;
varianza := varianza / (NoGood - 1); SD := sqrt(varianza);
SEmean := sqrt(varianza / NoGood);
fo i := 0 to N-1 do inicio
ValueStr := FloatToStr(X[i]);
si Trim(MissValue) = ValueStr entonces continue;
deviation := X[i] - Mean;
z[i] := deviation / SD;

devsqr := deviation * deviation; M2 := M2 + devsqr;
M3 := M3 + (deviation * devsqr);
M4 := M4 + (devsqr * devsqr);
fin fin
si NoGood > 3 entonces inicio
Skew := (NoGood * M3) / ((NoGood - 1) * (NoGood - 2) * SD * varianza);
num := 6.0 * NoGood * (NoGood - 1);
den := (NoGood - 2) * (NoGood + 1) * (NoGood + 3); SESkew := sqrt(num / den);
Kurtosis := (NoGood * (NoGood + 1) * M4) - (3.0 * M2 * M2 * (NoGood - 1)); Kurtosis := Kurtosis / ((NoGood - 1) * (NoGood - 2) * (NoGood - 3) *
(varianza * varianza));
SeKurtosis := sqrt((4.0 * (NoGood * NoGood - 1) * (SESkew * SESkew)) /
((NoGood - 3) * (NoGood + 5)));
fin
Result := NoGood;
fin
//------------------------------------------------------------------------ fin.
```

REFERENCIAS

David C Lay, Jesús Murrieta Murrieta (2007), Álgebra lineal y sus aplicaciones,

(Tercera Edición) Pearson Educación

David Moore (2006), Estadística Aplicada Básica, Antoni Bosch editor.

Dirk Louis, José Luis Cortés (2000), Delphi 5, Alfaomega Marcombo.

Francisco Charte Ojeda (2003), DELPHI 7 GUIA PRÁCTICA, Anaya Multimedia.

Gary Cornell (1996), Delphi para programadores, McGraw-Hill/Interamericana de

Espana, S.A.

Ian Marteens (2003), La cara oculta de Delphi 6, Danysoft Internacional.

Ibañez Quispe Vladimiro (1998), Modelos Lineales, Universidad Nacional del

Altiplano – Facultad de Ingeniería Estadística e Informática.

J.M. Juran, R.S. Bingham, Frank M. Gryna, José María Vallhonrat (1983), Manual de control de la calidad, Editorial Reverté.

Jaime Serret Moreno-Gil (1995), Manual de Estadística Universitaria, ESIC

Editorial.

F. Cortez Vásquez, Buenas Prácticas sobre Educación Indígena, 2014.