AICC103

**Taller 2**

**Representación del conocimiento**

**Nombres integrantes: Jonathan Lavín – Julio Neira**

**Número de grupo: 7**

**Fecha: 25-08-2024**

**Diagnóstico de Problemas de Computadora Parte 1**

1. **Introducción al Contexto del Problema**

El diagnóstico de problemas en computadoras es una tarea importante en el área informática, tanto para usuarios, como para técnicos de soporte. Identificar correctamente la causa de un mal funcionamiento puede evitar pérdidas de tiempo, proteger la integridad de los datos y prevenir daños permanentes en el hardware. Nuestro programa "Diagnóstico" tiene como objetivo agilizar este proceso mediante un sistema experto que guía al usuario a través de una serie de preguntas para determinar la naturaleza del problema.

1. **Diseño General del Programa**

El programa está diseñado como un sistema basado en reglas. Se utiliza Prolog para implementar la lógica, aprovechando su capacidad para manejar hechos y reglas de manera eficiente. La elección de Prolog se debe a su idoneidad para sistemas basados en el conocimiento, donde la inferencia lógica es fundamental.

El programa se inicia con el predicado inicio, que limpia cualquier conocimiento previo y luego aplica las reglas de diagnóstico para identificar el problema. Cada diagnóstico se basa en la evaluación secuencial de diferentes predicados, que representan síntomas o condiciones observables en la computadora.

1. **Representación del Estado del Juego**

En este contexto, el "estado del juego" se refiere a las condiciones actuales de la computadora, como si enciende, muestra imagen, etc. Esta representación se maneja a través de la base de datos dinámica de Prolog, utilizando el predicado relacion/3, que almacena las respuestas del usuario. Este enfoque permite que el programa mantenga un registro del progreso en el diagnóstico y aplique las reglas en función del conocimiento acumulado.

Una ventaja clave de esta representación es la flexibilidad y simplicidad que ofrece Prolog al manejar hechos y relaciones dinámicamente. Comparado con otras opciones, como almacenar el estado en estructuras de datos complejas, la base de datos dinámica de Prolog permite una manipulación directa y fácil actualización del conocimiento en tiempo real.

1. **Representación del Conocimiento**

El conocimiento se representa en forma de reglas lógicas y hechos almacenados dinámicamente. Las reglas de diagnóstico, definidas como diagnosticar/1, encapsulan el conocimiento experto en un formato lógico que Prolog puede evaluar eficientemente. Por ejemplo, la regla:

prolog

Copy code

diagnosticar(problema\_fuente\_poder) :-

\+ enciende,

\+ muestra\_imagen.

indica que, si la computadora no enciende y no muestra imagen, el problema podría estar en la fuente de poder. Este enfoque permite una clara separación entre los hechos (respuestas del usuario) y las reglas (conocimientos expertos), facilitando el mantenimiento y la actualización del sistema.

Una ventaja sobre otros enfoques, como las redes bayesianas o las máquinas de soporte vectorial, es que Prolog permite codificar conocimiento experto de manera explícita y comprensible, sin necesidad de entrenar un modelo o manejar probabilidades, lo que es ideal para escenarios donde el conocimiento es claro y estructurado.

1. **Conclusiones**

El desempeño del programa es efectivo en la mayoría de los casos, logrando identificar correctamente problemas comunes en computadoras. Sin embargo, su éxito depende de la precisión del usuario para proporcionar respuestas precisas.

**Ejemplos de casos:**

* Si un usuario reporta que la computadora no enciende ni muestra imagen, el sistema correctamente diagnostica un posible problema con la fuente de poder.
* Si la computadora enciende, pero no hay conexión a internet y el sistema operativo carga correctamente, el sistema diagnostica un problema de red.

**Limitaciones y mejoras:**

El sistema podría no cubrir todos los posibles problemas de una computadora, especialmente si los síntomas son ambiguos o múltiples problemas ocurren simultáneamente. Para mejorar, se podría:

* **Incluir más reglas** que cubran una gama más amplia de problemas.
* **Implementar un mecanismo de retroalimentación**, donde el sistema pueda aprender de errores pasados en diagnósticos incorrectos.
* **Agregar una capa de probabilidades** para manejar la incertidumbre en el diagnóstico, lo cual podría mejorar la precisión en escenarios más complejos.

El programa "Diagnóstico" proporciona una base sólida para la identificación automatizada de problemas en computadoras, utilizando un enfoque lógico y estructurado que podría expandirse y mejorarse para abarcar más escenarios y ofrecer diagnósticos más precisos.

**Filtro de Spam utilizando Naive Bayes Parte 2**

1. **Introducción al contexto del problema**

El filtrado de spam es un problema crítico en la gestión de correos electrónicos, ya que un volumen considerable de mensajes no deseados afecta la productividad y la seguridad de los usuarios.

En este proyecto, se desarrolló un filtro de spam utilizando dos técnicas de selección de características: TF-IDF y Random Forest. Posteriormente, se emplearon modelos de Bernoulli y Multinomial Naive Bayes para clasificar los correos electrónicos como spam o no spam. El objetivo principal fue comparar el rendimiento de estas técnicas y modelos para determinar cuál ofrece la mayor precisión en la identificación de correos electrónicos no deseados.

1. **Diseño general del programa, técnicas aplicadas y justificación de su elección**

El diseño del programa siguió una estructura lógica que comenzó con el preprocesamiento de los correos electrónicos y la selección de características, seguida de la aplicación de modelos Naive Bayes para la clasificación. A continuación se describen las técnicas clave:

* **Preprocesamiento de datos:** Todos los correos se convirtieron a minúsculas, se eliminaron caracteres especiales y números, y se aplicó lematización utilizando el modelo de spaCy. Además, se excluyeron las stopwords, dejando solo las palabras más relevantes para la clasificación.
* **Selección de características:**
  + **TF-IDF (Term Frequency-Inverse Document Frequency):** Se utilizó para calcular la importancia relativa de cada palabra en función de su frecuencia en el cuerpo del mensaje. Esta técnica permitió seleccionar las palabras más relevantes limitando el número a 1000 características.
  + **Random Forest:** Se entrenó un modelo de Random Forest para identificar las palabras más relevantes en función de su capacidad predictiva para clasificar correos como spam o no spam.
* **Modelos Naive Bayes:**
  + **Bernoulli Naive Bayes:** Se seleccionó debido a su capacidad para trabajar con datos binarios, es decir, la presencia o ausencia de palabras en los correos electrónicos.
  + **Multinomial Naive Bayes:** Este modelo fue elegido debido a su habilidad para manejar la frecuencia de palabras, lo que es crucial para clasificar textos con diferentes patrones de palabras.

La justificación de la elección de estas técnicas se basó en su eficiencia y capacidad comprobada para manejar grandes volúmenes de datos textuales de manera rápida y efectiva.

1. **Representación escogida para el estado del juego**

La representación de los correos electrónicos se hizo mediante dos enfoques diferentes:

* **TF-IDF:** Este enfoque pondera las palabras en función de su importancia en el cuerpo del mensaje, dándole más peso a las palabras distintivas y menos a las comunes. Este método fue efectivo para capturar las palabras más influyentes en la clasificación de spam.
* **Count Vectorizer:** Se utilizó junto con las palabras seleccionadas por Random Forest para representar los correos como vectores de frecuencia. Esta representación es adecuada para el modelo Multinomial Naive Bayes, que se basa en la frecuencia de aparición de las palabras.

1. **Representación escogida para el conocimiento**

Los modelos Naive Bayes manejaron la información textual de manera diferente:

* **Bernoulli Naive Bayes:** La representación del conocimiento en este modelo se basó en la presencia o ausencia de palabras, lo cual simplificó el procesamiento al enfocarse en patrones binarios. Esto resultó ventajoso para mensajes cortos y directos.
* **Multinomial Naive Bayes:** Este modelo aprovechó la información de la frecuencia de palabras, lo cual le permitió capturar matices más finos en el lenguaje de los correos electrónicos, especialmente en aquellos con una mayor cantidad de texto.

1. **Conclusiones**

**Desempeño y ejemplos**

Después de aplicar los modelos Bernoulli y Multinomial Naive Bayes a los correos electrónicos vectorizados mediante TF-IDF y Count Vectorizer, se observaron los siguientes resultados:

* **Mejor modelo Bernoulli Naive Bayes:** Este modelo alcanzó su mejor rendimiento utilizando 676 palabras seleccionadas por TF-IDF, logrando una precisión del 98.29%. Fue efectivo para clasificar mensajes cortos y directos, donde la presencia o ausencia de palabras clave fue determinante.
* **Mejor modelo Multinomial Naive Bayes:** Utilizando 136 palabras seleccionadas por Random Forest y vectorizadas con Count Vectorizer, este modelo alcanzó una precisión del 97.66%. Fue particularmente eficaz en correos con mayor cantidad de texto y múltiples repeticiones de palabras clave.

**Identificación correcta y áreas de mejora**

En general, ambos modelos lograron una alta precisión en la identificación de correos electrónicos. Sin embargo, se presentaron algunos falsos positivos y falsos negativos. Estos errores podrían minimizarse mejorando el preprocesamiento de los datos, ajustando los hiperparámetros de los modelos, o combinando Naive Bayes con otras técnicas como SVM o redes neuronales para mejorar la capacidad predictiva.

**Ejemplos de desempeño**

En algunos casos, el modelo presentó dificultades para clasificar correctamente correos electrónicos promocionales que, aunque deberían ser clasificados como spam, tienen características similares a los mensajes deseados. Este tipo de correos, debido a su contenido ambiguo, pueden llevar a errores de clasificación, ya que utilizan un lenguaje que puede confundirse fácilmente con el de correos legítimos. Esto subraya la necesidad de mejorar la diferenciación entre correos promocionales considerados como spam y aquellos que son verdaderamente deseados.

El modelo Bernoulli Naive Bayes, utilizando palabras seleccionadas y vectorizadas con TF-IDF, demostró ser el más preciso. Sin embargo, el modelo Multinomial Naive Bayes, combinado con la selección de características mediante Random Forest y la vectorización con Count Vectorizer, mostró un excelente equilibrio entre precisión y eficiencia. Ambos modelos son adecuados para la clasificación de spam, y futuras mejoras en la selección de características y la optimización de los hiperparámetros podrían llevar a un rendimiento aún mayor.

**Gráfico de palabras más comunes en correos spam y no spam**

Gráfico, Gráfico de barras

Descripción generada automáticamente

Gráfico, Histograma

Descripción generada automáticamente

**Gráficos de importancia acumulada**

Gráfico, Gráfico de líneas, Gráfico de dispersión

Descripción generada automáticamente

TF-IDF: Se necesitan 327 palabras para capturar el 70.0% de la importancia acumulada.

TF-IDF: Se necesitan 469 palabras para capturar el 80.0% de la importancia acumulada.

TF-IDF: Se necesitan 676 palabras para capturar el 90.0% de la importancia acumulada.

Gráfico

Descripción generada automáticamente

* Random Forest: 78 palabras para capturar el 70.0% de la importancia acumulada.
* Random Forest: 139 palabras para capturar el 80.0% de la importancia acumulada.
* Random Forest: 263 palabras para capturar el 90.0% de la importancia acumulada

**Gráficos de precisión según número de palabras**

Gráfico, Gráfico de líneas

Descripción generada automáticamente

**Gráficos de evaluación de acuerdo con importancia acumulada.**

Gráfico, Gráfico de líneas, Gráfico de dispersión

Descripción generada automáticamente

Gráfico, Gráfico de líneas

Descripción generada automáticamente

**Matrices de confusión**

Gráfico

Descripción generada automáticamente