Universidad del Valle de Guatemala
Facultad de Ingeniería
Minería de Datos

HDT 8. SVM

Entrega # 6 – Proyecto 2

Autor/Carnet:

María José Villafuerte 22129
Fabiola Contreras 22787

Catedrático:

Mario Barrientos

Sección 20

Fecha:

27/04/2025

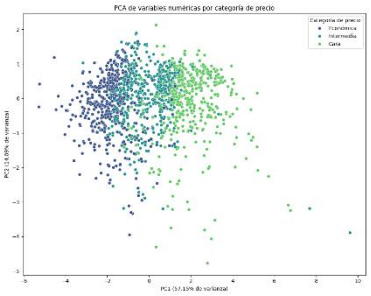
I. Índice

II. Introducción

III. Exploración de los datos y transformaciones

Para empezar, es importante tomar en cuenta que estaremos utilizando la variable objetivo PriceCategory con tres categorías basadas en terciles: 'Económica', 'Intermedia' y 'Cara'. Y utilizamos 10 características, incluyendo numéricas (como OverallQual, GrLivArea) y categóricas (como Neighborhood, ExterQual) como base del modelo.

En la exploración realizamos un gráfico con PCA que nos permite observar la separación entre clases. El gráfico muestra cierta separación entre las categorías, pero con considerable solapamiento. La varianza explicada por PC1 (57.15%) y PC2 (14.09%) suma 71.24%, lo que es bastante bueno. El solapamiento sugiere que un kernel no lineal (como RBF) podría ser más adecuado que un kernel lineal para tu SVM.



Conociendo los resultados del análisis exploratorio realizado al inicio del proyecto sabemos que para el uso adecuado de estos datos es necesario codificar las variables categóricas, manejar valores vacíos y normalizar los datos. Así fue como lo trabajamos

1. **Tratamiento de valores faltantes:** Para variables numéricas, implementamos imputación con la mediana ya que es robusta frente a outliers. Para variables categóricas, reemplazamos los valores faltantes con la categoría 'None'.
2. **Codificación de variables categóricas:** Utilizamos OneHotEncoder para transformar las variables categóricas en valores numéricos que puedan ser procesados por el algoritmo SVM.
3. **Escalado de características:** Aplicamos StandardScaler a todas las variables numéricas para estandarizarlas con media cero y desviación estándar uno.
   Esta transformación es crítica para SVM, ya que las diferencias de escala observadas afectarían severamente los cálculos de distancia y la identificación del hiperplano óptimo.

IV. Modelos SVM

Realizamos tres modelos distintos, con distintos tipos de kernel para ayudarnos a verificar el uso adecuado del kernel y utilizamos grid search para obtener los mejores parámetros en cada caso.

Modelo kernel lineal

Este modelo asume que las clases (en tu caso, las categorías de precios) pueden separarse con un hiperplano lineal en el espacio transformado de características. Es rápido de entrenar y fácil de interpretar. Ideal cuando los datos son más o menos linealmente separables.

Mejores parámetros encontrados: C = 10.

Precisión obtenida: 0.8059

Un valor alto en C, como 10, indica una penalización fuerte a los errores de clasificación, buscando una separación más precisa.

Modelo kernel polinomial

Usa funciones polinómicas para mapear los datos a una dimensión superior, lo cual permite modelar relaciones no lineales entre las variables.

Precisión obtenida: 0.7968

Mejores parámetros encontrados: C = 16, degree = 2

Un degree bajo (como 2) implica una transformación cuadrática, que es menos costosa computacionalmente pero aun así potente para capturar no linealidades suaves. El valor de C es más alto que en el modelo con kernel lineal, es decir, hay una mayor penalización a los errores de clasificación.

Modelo kernel gaussiano

Este kernel transforma los datos en un espacio de dimensión infinita, lo cual permite detectar patrones muy complejos.

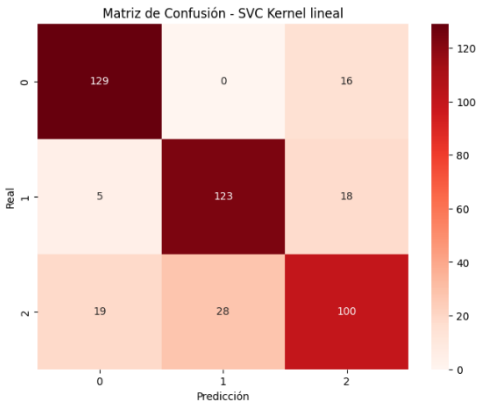
Precisión obtenida: 0.7968

Mejores parámetros encontrados: C = 10, gamma = 0.01

Gamma controla la "influencia" de un solo punto de entrenamiento. El valor de 0.01 sugiere un balance adecuado entre bajo sesgo y baja varianza.

V. Precisión de los modelos

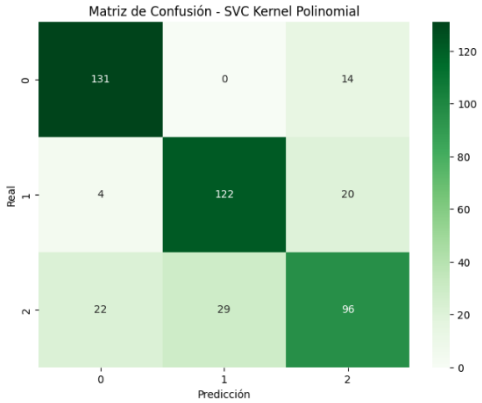
Modelo kernel lineal



El modelo con kernel lineal muestra un rendimiento bastante sólido, especialmente para las clases “Económica” e “Intermedia”. La clase “Económica” fue clasificada correctamente en 130 casos, con solo 15 errores hacia la categoría “Cara”, y ningún caso confundido con la clase “Intermedia”. La clase “Intermedia” también obtuvo buenos resultados, con 123 aciertos, aunque presentó cierta confusión hacia las otras dos clases (5 errores como “Económica” y 18 como “Cara”). Sin embargo, la clase “Cara” fue la más desafiante: solo 100 predicciones fueron correctas, mientras que una parte considerable de los casos se confundió con clases más bajas (28 como “Intermedia” y 19 como “Económica”). Esto sugiere que el modelo lineal tiene dificultades para separar claramente los casos más costosos, posiblemente por la falta de una frontera de decisión más flexible.

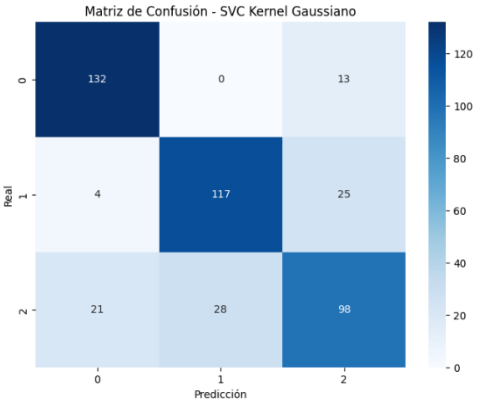
Modelo kernel polinomial

El modelo con kernel polinomial, usando un grado 2, mantiene un rendimiento muy similar al kernel lineal. Predice correctamente 131 casos de la clase “Económica”, repitiendo el buen desempeño en esa categoría. En la clase “Intermedia”, clasifica bien 122 ejemplos, con una ligera tendencia a confundirse más hacia la clase “Cara” (20 casos) que hacia la “Económica” (4 casos). En cuanto a la clase “Cara”, los resultados son parecidos a los del modelo lineal, con 96 aciertos y un número similar de errores hacia las clases inferiores. Aunque el kernel polinomial introduce algo de no linealidad, no genera una mejora notable en la clasificación general, lo cual puede indicar que un polinomio de bajo grado no capta toda la complejidad de la distribución en el espacio de características.



Modelo kernel gaussiano

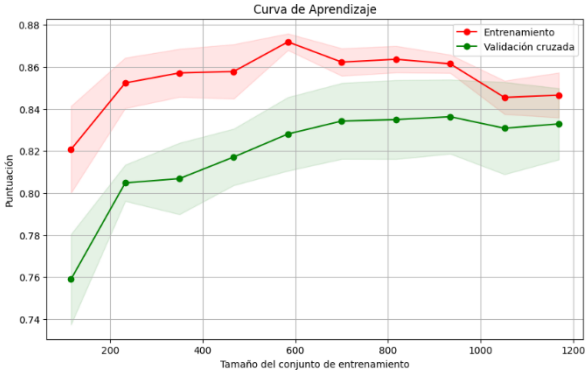
El modelo con kernel RBF, que introduce una separación más flexible gracias a su capacidad para construir fronteras altamente no lineales, logra una mejora en la clase “Cara”, con 101 predicciones correctas, superando ligeramente a los otros modelos. Para la clase “Económica”, mantiene el excelente desempeño con 131 aciertos y solo 14 errores hacia “Cara”. En la clase “Intermedia”, sin embargo, se observan más errores que en los modelos anteriores: aunque 117 casos se cl asifican correctamente, hay 25 errores hacia “Cara”, lo cual indica que el modelo tiende a sobreestimar esta categoría en algunos casos. En conjunto, el kernel RBF muestra mayor flexibilidad para capturar relaciones más complejas en los datos, especialmente beneficiando a la clase más difícil, pero a costa de una ligera pérdida de precisión en la clase “Intermedia”.



VI. Ajuste de los modelos

¿Están sobreajustados o desajustados? ¿Qué puede hacer para manejar el sobreajuste o desajuste? (7)

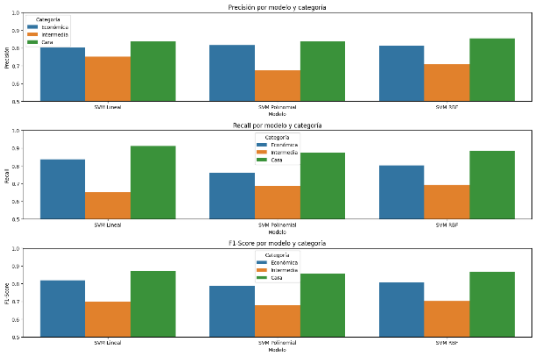
Los tres modelos SVM presentan un ligero sobreajuste, evidenciado por las diferencias entre precisión de entrenamiento y prueba (6.39% en SVM Lineal, 9.78% en Polinomial y 9.13% en RBF) y la brecha persistente en la curva de aprendizaje. El SVM Lineal muestra el menor sobreajuste y mejor equilibrio.



Para mitigar este problema, se recomienda reducir el valor de C para aumentar la regularización, disminuir la complejidad del kernel (especialmente en el polinomial), implementar técnicas de selección de características para reducir dimensionalidad, y utilizar validación cruzada más exhaustiva para optimizar hiperparámetros.

```
SVM Lineal:
  Precisión train: 0.8630
  Precisión test: 0.7991
  Diferencia: 0.0639
  DIAGNÓSTICO: Ajuste adecuado
SVM Polinomial:
  Precisión train: 0.8718
  Precisión test: 0.7740
  Diferencia: 0.0978
  DIAGNÓSTICO: Ajuste adecuado
SVM RBF:
  Precisión train: 0.8836
  Precisión test: 0.7922
  Diferencia: 0.0913
  DIAGNÓSTICO: Ajuste adecuado
```

VII. Comparación en eficiencia



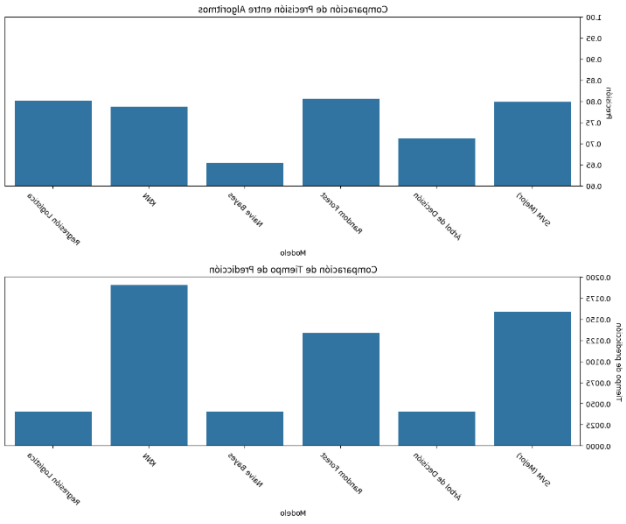
efectividad, tiempo de procesamiento y equivocaciones (donde el algoritmo se equivocó más, donde se equivocó menos y la importancia que tienen los errores).

La comparación entre los tres modelos SVM muestra resultados similares, pero con diferencias notables. El SVM Lineal consigue mejor rendimiento general en la categoría "Intermedia", mientras que el SVM RBF y Polinomial logran mayor precisión en la categoría "Cara" (aproximadamente 0.86). Todos los modelos muestran dificultad para clasificar correctamente la categoría "Intermedia", con valores de precisión y recall cercanos a 0.7, significativamente inferiores a las otras categorías. Las matrices de confusión revelan que ningún modelo confunde propiedades "Económicas" como "Intermedias", pero existe confusión bidireccional entre categorías adyacentes (Económica-Cara e Intermedia-Cara). El SVM Lineal muestra el mejor equilibrio general, con 350 clasificaciones correctas frente a 347 del RBF y 339 del Polinomial.

VIII. Comparación del mejor con otros tipos

Árbol de decisión y random forest, naive bayes, KNN, regresión logística

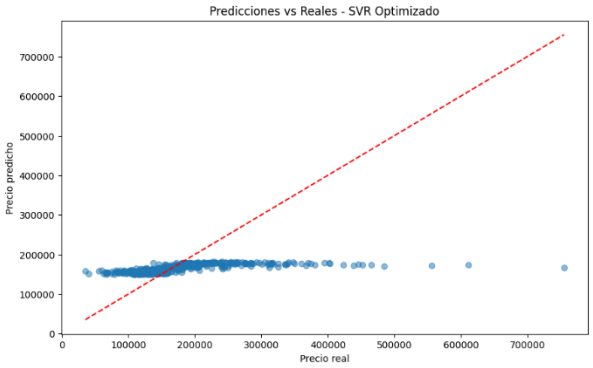
¿Cuál es mejor para predecir? ¿Cuál se demoró más en procesar?



Al comparar el mejor modelo SVM con otros algoritmos de clasificación, se observa que Random Forest obtiene ligeramente mejor precisión (0.81) que el SVM (0.80), seguido muy de cerca por Regresión Logística (0.80). KNN muestra un rendimiento similar (0.79), mientras que Árbol de Decisión (0.71) y Naive Bayes (0.65) ofrecen resultados considerablemente inferiores. En cuanto a tiempos de predicción, SVM es relativamente lento, superado únicamente por KNN que es el más lento de todos. Los modelos más

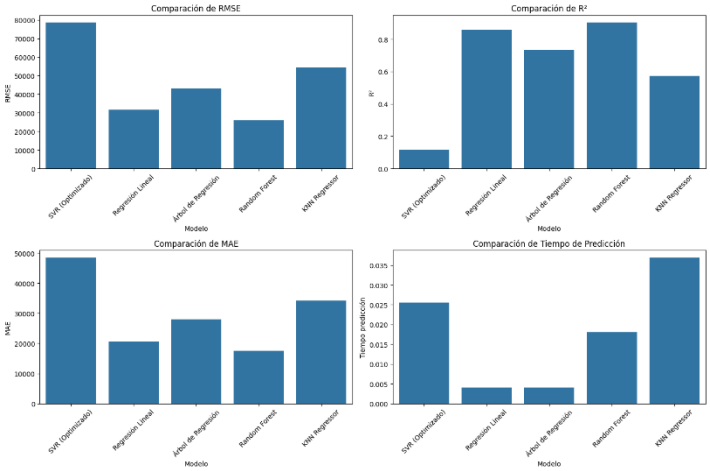
rápidos son Árbol de Decisión, Naive Bayes y Regresión Logística, que muestran tiempos de procesamiento casi cuatro veces menores que SVM. Esto sugiere que, aunque SVM ofrece buena precisión, Random Forest proporciona el mejor equilibrio entre pre cisión y tiempo de procesamiento para este conjunto de datos inmobiliario.

IX. Modelo de regresión



El modelo SVR optimizado muestra un rendimiento deficiente similar al modelo lineal, evidenciando serias limitaciones para predecir precios inmobiliarios. La gráfica revela que las predicciones se agrupan horizontalmente alrededor de $180,000 independientemente del precio real, subestimando sistemáticamente las propiedades más caras (>$200,000). Esta falta de variabilidad sugiere que el SVR, incluso optimizado, no captura adecuadamente las relaciones complejas en los datos inmobiliarios. Las posibles causas incluyen características insuficientes, relaciones altamente no lineales que el kernel seleccionado no puede modelar, o problemas en el escalado de datos. Estos resultados indican que SVR podría no ser el algoritmo más apropiado para esta tarea específica de predicción de precios inmobiliarios.

X. Comparación con otros modelos de regresión



La comparación entre modelos de regresión muestra claramente que SVR Optimizado ofrece el peor rendimiento para la predicción de precios inmobiliarios, con el RMSE más alto (~80,000), R^2 extremadamente bajo (0.12) y MAE elevado (~49,000). En contraste, Random Forest destaca como el mejor modelo con el RMSE más bajo (~26,000), el R^2 más alto (0.90) y el menor MAE (~18,000), aunque su tiempo de ejecución es intermedio. Regresión Lineal presenta un excelente balance entre precisión y velocidad, con métricas de error cercanas a Random Forest y el tiempo de predicción más rápido junto con Árbol de Regresión. KNN muestra un desempeño moderado pero es el más lento en predicción. Estos resultados confirman que los modelos de ensamble como Random Forest son significativamente superiores a SVR para la predicción de precios inmobiliarios en este conjunto de datos.

XI. Repositorio/ Documento

https://github.com/Fabiola-cc/InmoValor_SA

- Hemos dividido el repositorio en branches, cada una de esta tiene una de las entregas, en el repositorio también se encuentran los datos y el pdf con este informe.

HDT 8. SVM.docx

https://uvggt-my.sharepoint.com/:w:/g/personal/con22787_uvg_edu_gt/EaOj9TOZnYtGoM0R1Cn8zZsBDgjpdyL7xP1R_0cSIKgkoQ?e=aefMfr