03/04/23 - **¿Qué es la regularización y cómo aplicarla?**

|  |  |
| --- | --- |
| Puntos clave | Notas |
| La norma se usa en machine learning para encontrar errores.  **La norma L0**: devuelve la cantidad de elementos diferentes de 0 del vector.  **La norma L1**: devuelve la suma de los valores absolutos de los elementos del vector.  **La norma L2**: La magnitud del vector desde su origen.  **La norma infinita**: devuelve el valor absoluto más grande del vector  Un aporte adicional es que las normas se usan para determinar el módulo de regularización en la construcción de un modelo.  La regularización se usa para castigar la complejidad de un modelo y evitar overfitting, las más comunes son **L2** para simplicidad y **L1** para esparcidad. | Qué es la regularización y cómo aplicarla?  Esta técnica consiste en disminuir la complejidad de nuestro modelo a través de una penalización aplicada a sus variables más irrelevantes.     * Como podemos apreciar en la gráfica 1, hay un sub-ajuste, ya que la línea roja se acopla muy bien para los datos de prueba, pero no para los datos de entrenamiento. La línea roja en los datos de prueba da una mala generalización, una mala aproximación. * Entonces, la regularización consiste en introducir un poco de sesgo para introducir la varianza de los datos. * Pero para poder aplicar regularización necesitamos un término adicional el concepto de perdida. El concepto de perdida nos dice que tan lejos están nuestras predicciones de los datos reales, esto quiere decir que entre menor sea la perdida mejor será nuestro modelo.     Como podemos ver en la gráfica que la perdida tiende a disminuir, porque en algún momento van a ser vistos, van a ser operados y el modelo va a tender a ajustarse a esos datos de entrenamiento, pero lo que tenemos que mirar es cómo se va a comportar en el mundo real. |
| Sumario: | |

03/04/23 -

|  |  |
| --- | --- |
| Puntos clave | Notas |
|  | En el conjunto de validación o pruebas es muy normal que nuestra perdida comience a disminuir porque hay una buena generalización, pero llega un punto donde nuevos valores comienza a introducirse donde esa perdida vuelve a comenzar a subir ese es el punto donde en general se considera que comienza a haber sobreajuste. Es la perdida la medida que vamos a utilizar para poder utilizar la regularización.  **¿Cuántos tipos de regularización existen?**  **L1 Lasso:** Reducir la complejidad a través de eliminación de features que no aportan demasiado al modelo.  Penaliza a los features que aporta poca información volviéndolos cero, eliminado el ruido que producen en el modelo.    **L2 Ridge:** Reducir la complejidad disminuyendo el impacto de ciertos features a nuestro modelo.  Penaliza los features poco relevantes, pero no los vuelve cero. Solamente limita la información que aportan a nuestro modelo.    **ElasticNet:** Es una combinación de las dos anteriores.  **Lasso vs Ridge**   1. No hay un campeón definitivo para todos los problemas. 2. Si hay pocos features que se relacionen directamente con la variable a predecir: Probar Lasso. 3. Si hay varios features relacionados con la variable a predecir: Probar Ridge.   **ElasticNet: Una técnica intermedia**  He podido ver dos técnicas de regularización en las cuales añadimos un componente de penalización en el proceso donde encontramos los valores de los parámetros |
| Sumario: | |

03/04/23 - **ElasticNet: Una técnica intermedia**

|  |  |
| --- | --- |
| Key Points | Notes |
|  | 𝛽 minimizando la función de error.  Por ejemplo, si usamos el método de Mínimos Cuadrados Ordinarios, tenemos por definición nuestra función definida como:    Ahora bien. Si aplicamos la regularización L1 también conocida como Lasso (Least Absolute Shrinkage and Selection Operator), tenemos una ecuación de la forma:    donde tenemos un parámetro de ajuste llamado ƛ que si tiene valores altos para el problema mandará el valor de 𝛽j a 0.  Por otro lado. Si aplicamos la regularización L2 también conocida como Ridge, tendremos la siguiente ecuación:    Tendremos una penalización también pero que no tiene la posibilidad de llevar los valores de los coeficientes a cero. Sin embargo esto nos permitirá realizar el intercambio de +sesgo por -varianza.  Recordando que :   1. Ninguna de las dos es mejor que la otra para todos los casos. 2. Lasso envía algunos coeficientes a cero permitiendo así seleccionar variables significativas para el modelo. 3. Lasso funciona mejor si tenemos pocos predictores que influyen sobre el modelo. 4. Ridge funciona mejor si es el caso contrario y tenemos una gran cantidad. |
| Sumario: | |

03/05/23 -

|  |  |
| --- | --- |
| Key Points | Notes |
|  | Para aplicarlos y decidir cuál es el mejor en la práctica, podemos probar usando alguna técnica como cross-validation iterativamente. o bien, podemos combinarlos  **Regularización ElasticNet**  Es común encontrarnos en la literatura con un camino intermedio llamado ElasticNet. Esta técnica consiste en combinar las dos penalizaciones anteriores en una sola función. Así, nuestra ecuación de optimización quedará:    Donde tenemos ahora un parámetro adicional 𝛂 que tiene un rango de valores entre 0 y 1. Si 𝛂 = 0 , ElasticNet se comportará como Ridge, y si 𝛂 = 1 , se comportará como Lasso. Por lo tanto, nos brinda todo el espectro lineal de posibles combinaciones entre estos dos extremos.   1. Tenemos una forma de probar ambas L1 y L2 al tiempo sin perder información. 2. Supera las limitaciones individuales de ellas. 3. Si hace falta experiencia, o el conocimiento matemático de fondo, puede ser la opción preferente para probar la regularización.   **ElasticNet con Scikit-learn**   * Para implementar esta técnica añadimos primero el algoritmo ubicado en el módulo linear\_model. * from sklearn.linear\_model import ElasticNet * Y luego simplemente lo inicializamos con el constructor ElasticNet() y entrenamos con la función fit(). * regr = ElasticNet(random\_state=0) * regr.fit(X, y)   **El problema de los valores atípicos**  Un valor atípico es cualquier medición que se encuentre por fuera del comportamiento general de una muestra de datos.  Pueden indicar variabilidad, errores de medición o novedades. |
| Sumario: | |

03/05/23 - El problema de los valores atípicos

|  |  |
| --- | --- |
| Key Points | Notes |
|  | **Por qué son problemáticos**?  Pueden generar sesgos importantes en los modelos de ML.  A veces contienen información relevante sobre la naturaleza de los datos.  Detección temprana de fallos.  **Cómo identificarlos**?  **A través de métodos estadísticos**:   1. Z - Score: Mide la distancia (en desviaciones estándar) de un punto dado a la media. 2. Técnicas de clustering como DBSCAN. 3. Si q< Q1-1.5IQR ó q > Q3+1.5IQR     **Random Sample Consensus (Ransac)**: selecciona una muestra aleatoria de los datos asumiendo que esa muestra se encuentra dentro de los valores inliners, con estos datos se entrena el modelo y se compara su comportamiento con respecto a los otros datos. Este procedimiento se repite tantas veces como se indique y al finalizar el algoritmo escoge la combinación de datos que tenga la mejor cantidad de inliners, donde los valores atípicos puedan ser discriminados de forma efectiva. Ejemplo: |
| Sumario: | |

03/05/23 - **Regresiones Robustas en Scikit-learn & Qué son los métodos de ensamble?**

|  |  |
| --- | --- |
| Key Points | Notes |
| **Un proceso de ML tiene 3 partes**   * Limpiar el data sets. * entrenar modelos y ver cuáles son los mejores. * Ensamblar los mejores modelos.   **Bagging**: Qué tal si en lugar de depender de la opinión de un solo "experto" consultamos la opinión de varios expertos en paralelo e intentamos  lograr un consenso?  Modelos Ensamblados basados en Bagging:   * Random Forest. * Voting Classifiers/Regressors. * En general se puede aplicar sobre cualquier   familia de modelos de Machine Learning.  **Boosting**:  Es un método secuencial.  Busca fortalecer gradualmente un modelo de aprendizaje usando siempre el error residual de las etapas anteriores. | **Huber Reggresor**: no elimina los valores atípicos sino que los penaliza. Realiza el entrenamiento y si el error absoluto de la perdida alcanza cierto umbral (epsilon) los datos son tratados como atípicos. El valor por defecto de epsilon es 1.35 ya que se ha demostrado que logra un 95% de eficiencia estadística.  **Qué son los métodos de ensamble**?  Preview  No hay un ganador absoluto; depende de los datos, la simulación y las circunstancias.  El “embolsado” **Bootstrap agregattion(Bagging)** y el “aumento” **impulsar/propulsar (Boosting)** reducen la varianza de su estimación única, ya que combinan varias estimaciones de diferentes modelos. Así que el resultado puede ser un modelo con mayor estabilidad.  Si el problema es que el modelo único obtiene un rendimiento muy bajo, Rara vez el embolsado obtendrá un mejor sesgo. Sin embargo, Boosting podría generar un modelo combinado con menos errores ya que optimiza las ventajas y reduce las dificultades del modelo único.  Por el contrario, si la dificultad del modelo único es el ajuste excesivo, entonces el Bagging es la mejor opción. El refuerzo por su parte no ayuda a evitar el sobreajuste; de hecho, esta técnica se enfrenta a este problema en sí. Por esta razón, el Embolsado es efectivo con más frecuencia que el Impulso.  **Similitudes y Diferencias**  Ambos son métodos conjuntos para obtener N alumnos de 1 alumno... pero, si bien se construyen de forma independiente para Bagging, Boosting intenta agregar nuevos modelos que funcionan bien donde fallan los modelos anteriores. |
| Sumario: | |

03/05/23 – **Modelos de Ensamble y Bagging Meta-estimador**

|  |  |
| --- | --- |
| Puntos clave | Notas |
| * El resultado final también se consigue por consenso entre   todos los modelos. | Ambos generan varios conjuntos de datos de entrenamiento por muestreo aleatorio... pero solo Boosting determina los pesos de los datos para inclinar la balanza a favor de los casos más difíciles.  Ambos toman la decisión final promediando los N alumnos (o tomando la mayoría de ellos) pero es un promedio igualmente ponderado para Bagging y un promedio ponderado para Boosting, más peso para aquellos con un mejor rendimiento en los datos de entrenamiento.  Ambos son buenos para reducir la varianza y proporcionan una mayor estabilidad... pero solo Boosting intenta reducir el sesgo. Por otro lado, el embolsado puede resolver el problema de sobreajuste.  **Estrategias de clustering**  Los algoritmos de clustering son las estrategias que podemos usar para agrupar los datos de tal manera que todos los datos pertenecientes a un grupo sean lo más similares que sea posible entre sí, y lo más diferentes a los de otros grupos.  **Casos de aplicación de clustering**   1. No conocemos con anterioridad las etiquetas de nuestros datos (Aprendizaje no supervisado). 2. Queremos descubrir patrones ocultos a simple vista. 3. . Queremos identificar datos atípicos. |
| Sumario: | |

1/4/23 - **Validación de nuestro modelo usando Cross-Validation**

|  |  |
| --- | --- |
| Puntos clave | Notas |
|  | **Casos de uso de aplicación**:   1. Cuando sabemos cuántos grupos “k” queremos en nuestro resultado. Si es el caso, por ejemplo: en una empresa de marketing y sabemos que los segmentos de clientes son bajos, medio alto, en este caso es recomendable usar k-means, o bien, spectral clustering. 2. Cuando queremos que el algoritmo descubra la cantidad de grupos “k” óptima según los datos que tenemos. Por otro lado si no conocemos cuantos grupos o cuantas categorías tenemos y solo queremos experimentar, la solución puede ser Meanshift, clustering jerárquico o DBScan.   **Validación de nuestro modelo usando Cross-Validation**  **La última palabra siempre la van a tener los datos:** Todas nuestras intuiciones no tienen nada que hacer frente a lo que digan los datos y las matemáticas que aplicamos sobre estos datos. Por eso es importante siempre tener rigurosidad a la hora de evaluar los resultados que estamos recibiendo.  **Necesitamos mentalidad de testeo**: No se trata solamente de probar un poco al principio y un poco al final, sino que tendremos que probar constantemente durante todo el proceso, para poder encontrar cuál es la solución óptima que realmente nos soluciona el problema que tenemos pendiente, todo esto:   1. con varias formas 2. con varios conjuntos de datos 3. con varias configuraciones de parámetros 4. con varias distribuciones de nuestros datos   **Todos los modelos son malos, solamente algunos son útiles**  Todos los modelos que nosotros hacemos en últimas son una sobre simplificación de lo que pasa realmente. Entonces nunca nuestros modelos van a corresponder con la realidad al cien por ciento. Si jugamos lo suficiente y si somos lo suficientemente hábiles para configurar, vamos a llegar a un punto donde el modelo que estamos trabajando va a ser útil para ciertos casos específicos dentro del mundo real. |
| Sumario: | |

03/06/23 – Cross-Validation

|  |  |
| --- | --- |
| Puntos clave | Notas |
|  | **Tipos de validación**  **Hold-Out:** Se trata de dividir nuestros datos entrenamiento/pruebas, básicamente consiste en usar porcentajes fijos, por lo regular 70% de entrenamiento y 30% de pruebas.  hold-out hold-out-strategy  **Cuándo utilizar Hold-on?**   * Se requiere un prototipado rápido. * No se tiene mucho conocimiento en ML. * No se cuenta con abundante poder de cómputo.   **K-Folds**: Usar validación cursada K-Fold, aquí vamos a plegar nuestros datos k veces, el k es un parámetro que nosotros definimos y en esos pliegues vamos a utilizar diferentes partes de nuestro dataset como entrenamiento y como test, de tal manera que intentemos cubrir todos los datos de entrenamiento y de test, al finalizar el proceso.  k-fold k-fold-strtegy |
| Sumario: | |

03/06/23 -

|  |  |
| --- | --- |
| Puntos clave | Notas |
|  | **Cuándo utilizar K-Folds**?   * Recomendable en la mayoría de los casos. * Se cuenta con un equipo suficiente para desarrollar ML. * Se requiere la integración con técnicas de optimización paramétrica. * Se tiene más tiempo para las pruebas.   **LOOCV**: Validación cruzada LOOCV, **Leave One Out Cross Validation**. Este es el método más intensivo, ya que haremos una partición entre entrenamiento y pruebas, porque vamos a hacer entrenamiento con todos los datos, salvo 1 y vamos a repetir este proceso tantas veces hasta que todos los datos hayan sido probados.  loocv  **Cuándo utilizar LOOCV**?   * Se tiene gran poder de computo * Se cuentan con pocos datos para poder dividir por Train/Test * Cuando se quiere probar todos los casos posibles (para personas con TOC) |
| Sumario: | |