Capitulo 2 - Estado del arte

1. Aprendizaje supervisado

Dentro del área de aprendizaje de máquina el aprendizaje supervisado es una técnica que consiste en aprender una función a partir de un conjunto de datos de entrenamiento. El conjunto de entrenamiento consiste en pares de objetos de entrada y salida esperada. Utilizando estos datos se aprenderá una función que luego será capaz de predecir la salida esperada utilizando datos de entrada nunca vistos.

Las técnicas de aprendizaje supervisado pueden subdividirse en dos clases: clasificación, si la salida a predecir es discreta, y regresión cuando la salida es un valor continuo. A continuación describiremos una serie de métodos de aprendizaje supervisado los cuales son utilizados para tareas de regresión.

2. Métodos de regresión:

Regresión Lineal Simple:

El análisis de regresión de línea recta implica una variable de respuesta Y y una sola variable predictora X. Esta es la forma de regresión más simple que podemos encontrar, donde se modela Y como una función lineal de X. Esto es,

*y* = *b* + a*x*; (2-1)

donde la varianza de y es supuesta constante, b y a son los coeficientes de regresión que especifican la intercepción con el eje Y, y la pendiente de la recta respectivamente.

Estos coeficientes pueden ser resueltos mediante el método de mínimos cuadrados, el cual estima la recta de mejor ajuste como aquella que minimiza el error entre el dato verdadero y la estimación dada por la línea recta.

Sea |D| el conjunto de entrenamiento que contiene puntos de datos de la forma (, ), (, ),…, (, ), podemos definir el error en función de los parámetros como:

(2-2)

Donde , es el valor verdadero e el valor estimado por la función de regresión.

Queremos encontrar los valores de a y b que hagan minimicen la función de error. Diferenciando esta función con respecto a cada parámetro hayamos las siguientes ecuaciones las cuales nos proporcionan los parámetros de la recta de mejor ajuste:

(2-2)

(2-3)

Donde es el promedio de x1, x2, ..., x|D|, y el correspondiente a y1, y2, ..., |D|.

Ejemplo 1

Regresión lineal usando el método de mínimos cuadrados. La tabla 2-1 muestra los pares de datos: años de experiencia laboral de un graduado universitario(X) y el salario correspondiente(Y).

|  |  |
| --- | --- |
| años de experiencia (X) | salario(Y)(miles) |
| 3  8  9  13  3  6  11  21  1  16 | 30  57  64  72  36  43  59ª  90  20  83 |

Tabla 2-1

Figura 2-1

El grafico de la figura 2-1 sugiere una relación lineal entre las dos variables X e Y. De esta manera podemos modelar la relación entre el salario y la cantidad de años de experiencia mediante la ecuación de regresión lineal 2-1. Utilizando los datos de la tabla 1 junto con las ecuaciones (2-3) y (2-4) obtenemos el valor de los coeficientes:

a = 23,2

b = 3,5:

Reemplazando en 2-1 obtenemos la función de la recta de regresión:

Y = 23,6 + 3,5X.(2-4)

Utilizando esta función podemos predecir el valor de salario para un nivel de experiencia de la cual no tenemos información. Por ejemplo podemos decir que se estima que un egresado universitario con X = 10 años de experiencia, posee un salario de: Y = 23,6 + 3,5 \* 10.= $58600.

La regresión lineal múltiple es una extensión de la regresión lineal simple de manera de poder incorporar más de una variable predictora. La misma permite modelar la variable de respuesta como una función lineal de N variables predictoras, formando una tupla, ( X = (x1, x2, : : : , xn).) Nuestro conjunto de entrenamiento D contiene datos de la forma (X1, y1), (X2, y2), : : : , (XjDj, yjDj), donde las Xi son tuplas de entrenamiento N dimensionales, con etiquetas de clases asociadas Yi. Un ejemplo de un modelo de regresión múltiple basado en dos variables predictoras es:

Y = w0 + w1x1+w2x2, (2-5)

El método de cuadrados mínimos es extendido para encontrar los parámetros w0, w1, y w2 que minimicen la función de error asociada a este modelo. Las ecuaciones de regresión múltiple son tediosas para resolver a mano y este tipo de problemas son comúnmente resueltos usando paquetes estadísticos o de minería de datos.

La regresión lineal es un método simple pero poderoso para ser utilizado en la predicción numérica, el mismo ha sido usado ampliamente en aplicaciones estadísticas durante décadas. La desventaja que presenta este método es la linealidad. Cuando los datos exhiben una dependencia no lineal, la mejor recta de ajuste será encontrada, mediante el método de mínimos cuadrados. Esta recta no se ajustará demasiado bien a este tipo de datos. A pesar de esto los modelos lineales son interesantes ya que sirven como base para el desarrollo de métodos de aprendizaje más complejos.

b. Redes neurales para regresión

Una Red Neuronal Artificial (Red Neuronal o ANN), es un modelo computacional inspirado en las redes neuronales biológicas. La red está conformada de elementos de procesamiento (neuronas), conexiones entre los elementos de procesamiento y coeficientes (pesos) asociados a cada conexión. Todos estos elementos forman la estructura neuronal. Las redes neuronales, en su mayoría, tienen la particularidad de ser sistemas adaptativos, es decir que adaptan su comportamiento de acuerdo al ambiente en donde se encuentran. Así la red es capaz de modificar su estructura, en la fase de aprendizaje, de acuerdo a la información que se le presenta a la misma. En términos prácticos las redes neuronales son técnicas de modelamiento no lineales capaz de modelar funciones complejas. Estas pueden ser aplicadas a problemas de predicción, clasificación o control en un amplio espectro de campos como finanzas, neurociencia, medicina, ingeniería y física.

El bloque fundamental para la construcción de la red neuronal artificial es el modelo matemático de una neurona, como se muestra en la figura. Los tres componentes básicos de una neurona artificial son:

1. Las conexiones que proveen pesos Wj, a los valores de entrada Xj
2. Un sumador el cual que suma las entradas con sus respectivos pesos para computar el valor de entrada a la función de activación.
3. Una función de activación g que mapea v a g(v) el valor de salida de la neurona.



Figura 2-2

Mientras existen numerosas arquitecturas de redes neuronales, las aplicaciones con mejores resultados en el area de minería de datos han sido las redes multilayer feedfoward. Estas son redes en las cuales existe una capa de nodos que simplemente aceptan los valores de entrada y capas sucesivas que son neuronas como las de la figura 2-2. Las salidas de las neuronas en una capa son entrada a las neuronas en la capa sucesiva. La última capa es denominada capa de salida. Las capas entre la entrada y la salida son denominadas capas ocultas, ya que no interactúan con el medio externo. La figura 2-3 muestra un diagrama para esta arquitectura



Figura 2-3

En un contexto de aprendizaje supervisado para predicción numérica existe una sola neurona en la capa de salida cuya salida representa la predicción realizada a partir de los datos presentados en la capa de entrada.

Una de las características más importantes y que más interesante hace a las redes neuronales es su habilidad para aprender. Si consideramos una red neuronal como una función de mapeo F: X→Y, siendo X un vector de entrada a la red e Y un vector de salida de la misma. Dada una tarea específica para resolver y una clase de función F, aprender significa usar un conjunto de observaciones para encontrar un f\* Ȇ F, que resuelva la tarea específica de manera óptima.

Esta definición implica definir una función de costo C: F→R tal que para la solución optima f\*, C (f\*) <= C (f) para todo f Ȇ F. Es decir que ninguna otra solución tiene un costo menor al costo de la solución óptima. Los algoritmos de aprendizaje realizan una búsqueda en todo el espacio de soluciones para encontrar la solución que menor costo produce.

A pesar que la función de costo puede ser elegida de manera arbitraria, la elección de la misma suele ser realizada basada en las propiedades de la misma (convexidad) y también en las particularidades del problema que se intenta resolver. Finalmente la elección de la función de costo dependerá del tipo de tarea que intentemos resolver. La función de costo más utilizada en la práctica resulta ser la función de mínimos cuadrados. Esta resulta ser la misma función utilizada en la regresion lineal simple(Ver ecuación 2-2).

Entrenar una red neuronal significa encontrar una función f\* Ȇ F, siendo F: X→Y, tal que minimice el criterio de costo utilizado. El problema de entrenamiento puede dividirse en dos: aprender la estructura de la red y aprender los pesos de las conexiones. Existen numerosos algoritmos de entrenamiento, que resuelven de manera simple los valores de los pesos, dada una estructura de red fija. Ejemplo de ellos son: BackPropagation, Quick Propagation, [Conjugate Gradient Descent](http://en.wikipedia.org/wiki/Conjugate_gradient_method). Por otro lado si bien existen algoritmos para encontrar una estructura de red adecuada este aspecto del problema suele ser resuelto a través de la experimentación.

Los algoritmos de estimación de pesos son, en su mayoría, aplicaciones de la teoría de optimización y de estimación estadística. Estos utilizan alguna variante de la técnica de optimización [Gradient Descent](http://en.wikipedia.org/wiki/Gradient_descent). El algoritmo más conocido es denominado Back Propagation.

La técnica de Back Propagation se compone de un ciclo de dos fases: una fase hacia adelante, donde un dato de entrenamiento se introduce en la red y se calculan las salidas de todos los nodos hasta llegar al nodo final que produce el resultado de predicción. Una fase de retroceso en la cual se van actualizando los pesos de las conexiones desde de los nodos de la capa de salida hasta la capa de entrada.

Backpropagation utiliza Gradient Descent, esta técnica de optimización iterativa usa la información de la derivada de primer orden de la función de costo para ajustar los pesos de la red. A partir del valor de las derivadas, las multiplica por una pequeña constante llamada tasa de aprendizaje y luego sustrae el resultado al valor actual del peso. Esto es repetido en cada ciclo hasta que el cambio en el valor del peso se torna muy pequeño, de esta manera hemos encontrado la configuración de los pesos que logran un minimo de la función de costo elegida.

La taza de aprendizaje determina el incremento en dirección al mínimo y por lo tanto que tan rápido la búsqueda converge. Si esta taza es muy grande y la función tiene múltiples mínimos, la búsqueda puede pasar por alto algún mínimo, o puede oscilar fuertemente. Si la taza es pequeña el progreso hacia un mínimo puede volverse demasiado lento. Cabe destacar que el método de Gradient Descent solo puede encontrar un mínimo local. Si la función de costo tiene varios mínimos puede ser que no se encuentre el mínimo óptimo. Para aliviar este problema suelen realizarse múltiples corridas inicializando los valores de pesos en forma aleatoria.

Como en cualquier otra técnica de aprendizaje de maquina en las redes neuronales podemos sufrir el problema de sobre entrenamiento(overfitting), es decir que la red puede reflejar una buena perfomance con los datos de entrenamiento, pero no así con datos nunca vistos.

Early stopping es una modificación a la técnica de gradient descent la cual resulta en tener un conjunto de datos separado para verificar la perfomance de la red en cada iteración del ciclo de backpropagation. Cuando la performance medida con este conjunto de datos empieza a decaer, indicando sobre entrenamiento, el algoritmo es terminado.

Una pasada por todos los datos de entrenamiento se denomina una Epoca. La mayoría de las aplicaciones de las redes feedforward y de backpropagation requieren varias épocas antes de que los errores sean razonablemente pequeños.

El momentum es una solución para minimizar el número de épocas necesarias para encontrar un minimo aceptable. La misma consiste en agregar al peso que se está actualizando una proporción del incremento agregado en la iteración previa. Esto genera que el proceso de búsqueda sea más suave haciendo los cambios en dirección menos abruptos y favoreciendo una convergencia más rápida. Valores altos en el parámetro del momentum forzaran a que los ajustes sucesivos sean en direcciones similares. Otra idea es variar el parámetro de taza de aprendizaje para que este comience con un valor alto e ir decrementandolo a medida que se avanza de época.

Las redes neuronales pueden verse como una especie de sistema de procesamiento no lineal capaz de resolver un amplio espectro de problemas. Las redes neuronales son útiles cuando existen datos en abundancia pero se carece de una base teórica completa, es decir, no hay un modelo causal o una representación matemática. Los datos disponibles suelen ser no lineales, no estacionarios, o caóticos haciéndolos difíciles de modelar. Las redes neuronales no suponen ningún conocimiento previo acerca del espacio del problema, tampoco necesitan conocimientos previos en cuanto a la distribución estadística de los datos.

Las tareas en la cuales las redes neuronales son aplicadas se encuentran dentro de las siguientes categorías:[Aproximación de funciones](http://en.wikipedia.org/wiki/Function_approximation), o análisis de regresión, incluyendo predicción de series de tiempo y modelamiento. [Clasificación](http://en.wikipedia.org/wiki/Statistical_classification), incluyendo reconocimiento de patrones y reconocimiento de secuencias Procesamiento de datos, incluyendo filtrado, clustering y compresión.

Las áreas de aplicación de las redes neuronales incluyen: Sistemas de control(control de vehículos), juegos(backgammon, ajedres), reconocimiento de patrones(sistemas de radares, identificación de caras, reconocimiento de objetos), reconocimiento de secuencias(gestos, habla, escritura), diagnósticos médicos, aplicaciones financieras, descubrimiento de conocimiento en bases de datos, visualización y filtrado de email spam.

c. Árboles de decisión.

La regresión lineal es un modelo global, donde existe una única ecuación predictiva que se mantiene para todo el espacio de datos. Cuando los datos tienen muchas características los cuales interactúan de una forma complicada, en formas no lineales, encontrar un único modelo global puede ser muy difícil. Inclusive una vez encontrado este modelo, el mismo suele ser confuso. Una alternativa al enfoque no lineal es la de subdividir o particionar el espacio en regiones más pequeñas donde las interacciones son más accesibles. Luego estas particiones se vuelven a sub dividir y así sucesivamente. Finalmente obtenemos porciones del espacio en donde podemos utilizar modelos sencillos para encajar los datos. De esta manera el modelo global tiene dos partes: una consiste en la partición recursiva del espacio, la otra en aplicar un modelo simple a cada celda de la partición.

La alternativa mencionada no es más que la aplicación de la estrategia “Divide y vencerás”. Este enfoque conlleva a adoptar un estilo de representación de los datos en forma de árbol. En cada nodo de un árbol se evalúa un atributo en particular. Generalmente se compara el nodo con un atributo constante. Los nodos hojas dan una clasificación que se aplica a todas las instancias que alcanzan la hoja. Para clasificar una instancia desconocida, la misma es encaminada desde la raíz del árbol hacia abajo de acuerdo a los valores de los atributos que se evalúan en cada nodo y cuando una hoja es alcanzada la instancia es clasificada de acuerdo a la clase asignada para esa hoja.

Existen dos clases de arboles utilizados para predicción numérica, los arboles de regresión y los arboles modelo. La única diferencia entre ambos es que el primero almacena en sus nodos hojas el promedio de los datos que alcanzaron ese nodo, mientras que el segundo almacena un modelo de regresión lineal el cual es usado para predecir el valor de las instancias que alcanzan ese nodo. En lo que sigue describiremos los arboles modelos, ya que los arboles de regresión son un caso especial.

Construcción del árbol modelo.

La construcción del árbol modelo es un proceso recursivo. Comenzando del nodo raíz , se selecciona el atributo que mejor separa los datos de entrenamiento. Evaluando este atributo para cada instancia del conjunto de entrenamiento el mismo quedara separado en diferentes subconjuntos. Existirán tantos subconjuntos como posibles alternativas presente el nodo de decisión. Este proceso es repetido para cada subconjunto de los datos de entrenamiento hasta que todas las instancias que alcanzan un nodo tienen la misma clasificación.

Para determinar que atributo es el que mejor separa la porción T de los datos de entrenamiento que alcanzan un nodo en particular se utiliza el criterio de particionamiento. El mismo está basado en utilizar la desviación estándar de los valores de clase de T como una medida del error en ese nodo. El atributo que maximiza la reducción del error esperado es elegido para particionar los datos que llegan al nodo. La reducción del error esperado esta dado por la siguiente fórmula:



donde T1, T2, . . . son los conjuntos que resultan de separar el nodo de acuerdo al atributo elegido. El proceso de particionamiento termina cuando el valor de clase de las instancias que alcanzan un nodo varían muy poco, es decir cuando su desviación estándar es solo una pequeña fracción (Ej.: %5) de la desviación estándar del conjunto de instancias original. El particionamiento también termina cuando quedan unas pocas instancias en un nodo, por ejemplo: 4 instancias. La experimentación indica que los resultados obtenidos no son muy sensible al valor de estos parámetros.

Para predecir el valor de una instancia de prueba el árbol es atravesado hasta las hojas usando los valores de los atributos para decidir qué camino tomar en cada nodo. La hoja tendrá un modelo lineal el cual será utilizado para obtener el valor de la predicción. En vez de utilizar este valor directamente, resulta ser beneficioso realizar un proceso de suavizado para compensar las discontinuidades que inevitablemente ocurren entre los modelos adyacentes en las hojas del árbol. Este proceso es llevado a cabo implementando un modelo lineal en cada nodo interno del nodo, además del de las hojas. Luego una vez obtenida la predicción dada por el modelo en la hoja, este valor es filtrado durante el camino hacia el nodo raíz, suavizándolo en cada nodo que es atravesado mediante la combinación con el valor predicho por el modelo de cada nodo.

Un modelo apropiado para calcular este suavizado esta dado por:



“p’” es la predicción pasada al nodo superior en el árbol, “p” es la predicción pasada al nodo actual proveniente del nodo inferior, “q” es el valor predicho por el modelo en el nodo actual, “n” es el numero de instancias de entrenamiento que alcanzaron el nodo inferior, y “k” es una constante de suavizamiento.

La experimentación muestra que realizando este proceso de suavización se mejora substancialmente la precisión de las predicciones.

A pesar que los arboles construidos mediante el enfoque de “divide y vencerás” tienen un buen rendimiento sobre los datos de entrenamiento, los mismos suelen tener problemas de sobre entrenamiento y no pueden generalizar correctamente con conjuntos de pruebas independientes. Una solución a este problema es la realización de podas sobre el árbol.

Existen dos maneras para realizar la poda: Pre-poda y pos- poda. En pre-poda cuando se va construyendo el árbol y se encuentra una estructura que es lo suficientemente compleja, se detiene la construcción en esa rama. En pos-poda primero se construye todo el árbol y luego las descripciones complejas son extraídas

Pre-poda implica decidir durante la construcción del árbol, cuando parar de desarrollar los sub-arboles- esta particularidad resulta interesante ya que, a diferencia de pos-poda, no se hace trabajo de mas desarrollando sub arboles que luego podrían ser desechados. A pesar de esto pos-poda ofrece ciertas ventajas. Por ejemplo, existen situaciones en la que dos atributos considerados individualmente no tienen ningún aporte significativo, mientras que los mismos considerados en conjunto resultan ser muy informativos. De esta manera para asegurarnos de poseer la mayor información posible es necesario construir el árbol completo para luego desechar las partes que provocan sobre entrenamiento. La mayoría de los algoritmos de construcción de arboles utilizan pos-poda. Actualmente es una pregunta abierta si las estrategias de pre-poda pueden ser desarrolladas para alcanzar el mismo rendimiento que las estrategias de pos-poda.



Figura 2-4

En la figura 1 se muestra un ejemplo de un árbol modelo: En cada nodo se evalúa un atributo diferente y en las hojas se encuentra identificado el modelo lineal a utilizar.

*Maquinas de Soporte Vectorial para regresión :*

Las maquinas de soporte vectorial fueron desarrolladas como se conocen hoy en día en los laboratorios AT&T por Vappnik y colaboradores. Debido a este contexto industrial la investigación fue orientada a aplicaciones del mundo real, especificamente al reconocimiento de caracteres. En un corto periodo de tiempo estos clasificadores se convirtieron en competidores de las mejores técnicas existentes del momento. Así también las maquinas de soporte vectorial utilizadas para clasificación fueron extendidas para soportar problemas de regresión, obteniéndose también muy buenos resultados. Actualmente los algoritmos de maquinas de soporte vectorial forman parte de cualquier herramienta estándar para minería de datos. A continuación explicaremos los conceptos detrás de las maquinas de soporte de vectores utilizadas para regresión:

Como con la regresión lineal, la idea básica es encontrar una función que aproxime los puntos de entrenamiento minimizando el error en la predicción. La diferencia crucial es que todas las desviaciones hasta un parámetro ɛ dado son descartadas.

Un parámetro ɛ especificado por el usuario define un tubo alrededor de la función de regresión en los cuales los errores son ignorados: para soporte de vectores lineal el tubo es un cilindro. Si todos los puntos de entrenamiento caben dentro de un tubo de 2ɛ, el algoritmo obtiene una función en el medio del tubo más horizontal que los encierra. En este caso el error percibido es cero. La figura 2-5 muestra un problema de regresión con un atributo, una clase numérica, y ocho instancias. Es este caso el valor de ɛ fue configurado en 1, siendo el ancho del tubo alrededor de la función de regresión igual a 2. La figura 2-5 b muestra la salida del proceso de aprendizaje con el valor de ɛ configurado en 2. Como se puede apreciar un tubo más ancho hace posible aprender una función más horizontal.

El valor de ɛ controla que tan cerca la función encajaralos datos. Un valor demasiado grande producirá un predictor sin sentido – en el caso extremo, cuando 2ɛ excede el rango de valores de la clase de los datos de entrenamiento, la línea de regresión es horizontal y el algoritmo solo predice el valor promediode clase. Por otro lado, para valores pequeños de ɛ puede no haber un tubo que encierre todos los datos. En este caso algunos puntos de entrenamiento tendrán un error diferente de cero, y existirá un “trade-off” entre el error de predicción y la horizontalidad del tubo. En la **figura** **2-5a** ɛ fue configurado en 0.5 y no existe ningún tubo de ancho 1 que pueda encerrar todos los datos.

Para el caso lineal, la función de regresión con soporte de vectores puede ser escrita:

Los vectores de soporte son aquellos puntos que no caen estrictamente dentro del tubo –o sea, los puntos afuera del tubo y sobre el borde(Ver figura 1 a). Todos los puntos dentro del tubo se les asigna coeficiente 0 y pueden ser eliminados de los datos de entrenamiento sin cambiar la salida del proceso de aprendizaje.

En la mayoría de los casos los datos no son linealmente separables, son problemas no lineales, las MVS proveen soporte para este tipo de casos.

Cuando los datos no son linealmente separables en el espacio de entradas, son problemas no lineales; En este caso se puede realizar una transformación no lineal del espacio de entradas, en un espacio de características. Este nuevo espacio permite que los datos puedan ser separados linealmente de manera que se pueden aplicar los mismos razonamientos que para las MVS lineales. Asi podemos utilizar la ecuación de regresion 1, reemplazando el producto escalar por una función de nucleo. Una función núcleo o kernel se puede definir como aquella que permite realizar la separación y el traslado de los datos al espacio de características. Existen diversos kernels predeterminados conocidos entre los cuales se destaca el lineal, el RBF (Función de Base Radial), el polinomial, el sigmoidal, entre otros más.

El algoritmo de SVM para regresion funciona buscando simultáneamente una minimización del error y una maximización de la horizontalidad de la función de regresión. Sin embargo, cuando los datos no son linealmente separables incluso luego de aplicar una tranformacion, algunos puntos no encajan en el tubo**(Figu)** y no existe un tubo con error igual a 0. Estamos en presencia de un trade-off entre el error en la predicción y la horizontalidad del tubo. Este “trade-off” es controlado forzando un límite superior C en el valor absoluto de los coeficientes . El límite superior restringe la influencia de los vectores de soporte en la forma de la función de regresión y es un parámetro que el usuario debe especificar en adición a ɛ. Mientras más grande sea C lo más cerca la función **encajara(fit?)** los datos. En el caso degenerado (ɛ=0) el algoritmo simplemente realiza una regresión de *error absoluto minimo* utilizando la restricción del coeficiente y todas las instancias de entrenamiento se transforman en vectores de soporte. Contrariamente si ɛ es suficientemente grande como para que el tubo acomode todos los datos, el error se vuelve 0, no hay “trade-off” para hacer y el algoritmo obtiene el tubo mas horizontal que encierra a los datos indiferentemente del valor de C.

Comparado a otros métodos, como los arboles de decisión, incluso los algoritmos más veloces para soporte de vectores son lentos cuando son aplicados en un contexto no lineal. Por otro lado, suelen producir clasificadores muy precisos debido a los **detallados**(subtle) y complejos limites de decisión que pueden ser obtenidos.

**

**

*Figura 2-5*