Universitat Politècnica de Catalunya

Ciència i Enginyeria de Dades

Aprendizaje Automático 1: Comparación de algoritmos de clasificación binaria para el dataset Adult

Gerard Comas Quiles

June 2022

Índice

1. Introducción

Este proyecto se basa en aplicar todo el contenido de la asignatura en un problema real, creando un modelo (o varios) para entender como resolver este problema. El Machine Learning, o como nosotros lo llamaremos: Aprendizaje Automático, nos permite resolver tareas que serían prácticamente imposibles hacerlas de otra forma. Creando algoritmos capaces de "pensar por sí mismos"para reconocer patrones y extraer modelos a partir de esto. Es una área del conocimiento con muchísimo potencial y en cuanto se nos planteó este trabajo me paré a pensar en las infinitas opciones de aplicar estos algoritmos y conceptos.

Mi primera idea fue hacer un algoritmo capaz de detectar asteroides potencialmente peligrosos para la Tierra. La astrofísica es un campo que desde hace ya unos años me ha ido interesando cada vez más, es por eso, que durante el verano de 2019 colaboré en un par de proyectos con el Instituto de Astrofísica de Canarias. En ellos aprendí muchísimas cosas y entre ellas el concepto de los NEA, acrónimo en inglés de *Near Earth Asteroid*. Estos son los asteroides que tienen una órbita muy cercana a la Tierra, a menos de 0.3 ua, es decir un tercio de la distancia al Sol. El problema viene cuando estos asteroides intersecan su órbita con la de la Tierra, ya que esto quiere decir que en algún tiempo futuro podrían colisionar. Evidentemente esto es un problema serio incluso para la supervivencia de la raza humana y la NASA ya está tomando medidas de defensa.

Por la magnitud del problema y por el interés que me generaba, intenté enfocar el problema, pero mis esfuerzos fueron en vano pues el problema me sobrepaso. La idea inicial era que dada tres efemérides (posiciones del asteroide respecto la cúpula celeste) poder determinar si su órbita interaccionaría o no con la de la Tierra. El primer problema detrás de todo esto fue la base de datos, esta no estaba completo y solo me permitía saber para cada asteroide cuál era su órbita, pero el concepto de las efemérides en ninguna estaba implementado. Por lo que tendría que haber creado yo la base de datos. Encontré información sobre como hacerlo y de hecho no era tan complicado usando un script de Python, pero después me planteé si todo ese trabajo sería rentable. Descarté el problema debido a que posiblemente cualquier algoritmo simple con los que estamos tratando no podría generalizar orbitas a partir de tres puntos como era mi idea inicial. El conjunto de datos de entrada era grande, pero aun así las combinaciones de representar tres efemérides en la cúpula celeste son aún mayores. Me dio miedo que todo el esfuerzo no valiera la pena y decidí descartar el proyecto.

Sumido en la desesperación empecé a plantearme nuevos retos, y cada cual era más complicado que el anterior. Algunas de mis ideas fueron:

I) Reconocer una canción al estilo Shazam, pero solo una canción en concreto, por lo que el problema sería una clasificación binaria.

II) Predecir el resultado de una mano de Póker.

Pero estos problemas también fueron descartados. El primero debido a que esta vez tendría que hacer una base de datos y el problema se basaría en hacer análisis frecuencial, lo cual necesitaría un gran preprocesado de los datos. El segundo porque, de nuevo, no tenía base de datos y además me pareció que une estudio probabilístico obtendría mejor resultado. Por lo que decidí hacer caso a los profesores, en vez de encontrar un problema e intentar solucionarlo, limitarme a las bases de datos que tengo disponibles.

Después de repasarme todas las recopilaciones de bases de datos que nos proporcionaron los profesores no encontré ningún problema que realmente me llamará la atención, por lo que decidí centrarme en un problema que tuviera diversidad en los tipos de variables, que hiciera falta preprocesar un poco los datos y además me permitiera hacer varios estudios a la vez, aunque lo primordial será entender y analizar los algoritmos que emplearé.

Por lo tanto, seleccioné la base de datos de *Adult*, también conocida como *Census Income*. Es una extracción de la base de datos del censo estadounidense del 1994, hecha por *Barry Becker* y se puede descargar aquí: https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data

2. Base de datos

La idea de esta base de datos es predecir las ganancias anuales de una persona a partir de factores educativos, sociales y propiamente económicos de esta persona. De hecho más que predecir las ganancias anuales, será determinar si estas superan los 50.000 $ anuales o no. Convirtiéndolo en un problema de clasificación binaria. Este será el principal objetivo de la práctica y los algoritmos se basarán en solucionar esto.

Una vez ya he escogido el problema a solucionar, vamos a estudiar la base de datos en sí, para poder aplicar un preprocesado y depurar los datos. La base de datos consta de 32.561 instancias y 15 variables. Cuantos más datos, más precisos serán nuestros modelos y predicciones, pero siguiendo la recomendación de los profesores reduciré un poco la medida de los datos, ya que no dispongo ni de una gran RAM ni CPU. De hecho, lo más probable es que las dos clases no estén balanceadas de un inicio por lo que podremos utilizar esto a nuestro favor y reducir el número de datos mientras los balanceamos.

Como ya he comentado previamente, contamos con 15 variables de carácter muy distintas, desde variables categóricas binarias, hasta variables numéricas continuas e incluso variales categórica con gran cantidad de clases. La descripción de las variables la podremos encontrar en los Anexos, será importante la lectura de ese apartado para una mayor comprensión del análisis.

2.1. Preprocesado de datos

Entender bien nuestra base de datos será crucial para entender como enfocar el problema. Por lo que en esta sección, me dedicaré a depurar los datos para que se puedan analizar más fácilmente y podamos obtener mejores resultados.

2.1.1. Missing Values

Lo primero que hay que tener en cuenta son los *Missing Values*, es decir, valores sobre los cuales no se tiene información. En nuestro DataFrame solo hay missing values para tres variables: *workclass*, *ocupation* y *native-country*.

Si miramos el número de filas que contienen *Missing Values* vemos que son 2.399. Tenemos tres opciones para tratar estos datos:

1) Eliminar cualquier fila que contenga *Missing Values*

2) Eliminar las variables que contengan *Missing Values*

3) Predecir el valor de los *Missing Values*, teniendo en cuenta datos similares

Lo más cómodo y por lo que optaré será eliminar las filas en las que faltan valores, esto solo representa un 7.37 % de los datos. De hecho, ya nos interesa eliminar datos, pues como habíamos comentado interesa por limitaciones de Hardware analizar menos datos.

2.1.2. Variables irrelevantes

Nos podemos dar cuenta fácilmente que, la variable ***education*** y la variable ***education-num*** nos dan la misma información, de hecho, ***education-num*** nos da más información pues los valores numéricos más grandes indican un mayor nivel de estudios. Por lo que quitaremos ***education*** de nuestro DataFrame.

2.1.3. Transformación de las variables

Ahora miraremos en todas las variables categóricas como es la distribución de los datos en las diferentes clases, para ver si podemos agrupar clases para de esa forma simplificar más el posterior análisis. Esto resultará interesante sobre todo en la variable ***native-country***pues hay un número exagerado de clases.

Empezamos por ***workclass***, su distribución no es demasiado homogénea pues abunda la clase *Private*. Si hacemos un gráfico que represente las distribuciones según el ***Income*** para cada clase, vemos que las tres clases: *Federal-gov*, *Local-gov* y *State-gov*, tienen distribuciones muy parecidas entre ellas, con un 61.35 %, 70.52 % y 72.80 % de cobrar menos de 50K respectivamente. Por ese motivo, aunque los *Federal-gov* cobren un poco más en proporción, los agruparemos en una clase llamda *Gov*. Por último, podría parecer útil juntar las clases

Self-emp-inc y *Self-emp-not-inc* por su similitud en el nombre, pero como podemos ver en sus distribuciones no tienen nada que ver, con un 44.27 % y un 71.51 % de cobrar menos de 50K.

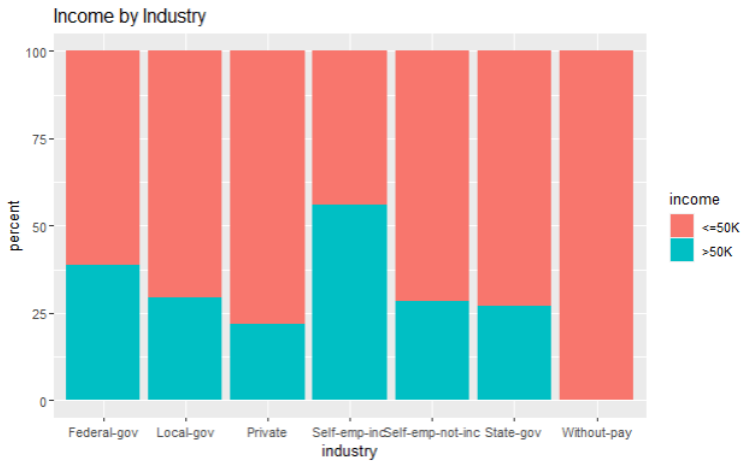

Figura 1: Histograma de la proporción entre los ***income*** para cada clase de ***workclass***

Podemos extraer de este gráfico que, por diferencia, quienes tienen mayor proporción de cobrar más de 50K son los de la clase *Self-emp-inc*.

Seguiremos con ***education-num***, aunque esté compuesto por valores numerables, cada número representa una clase, no se pueden hacer operaciones aritméticas entre ellos. De nuevo haremos un gráfico que nos muestre las proporciones para cada clase, claramente podemos ver que cuanto mayor sea el grado de estudio, más proporción hay de gente que cobra más de 500k. Agruparemos las clases del 1 al 8, pues son poco representativas y mantienen una distribución parecida, también agruparemos las clases 15 y 16, ya que prácticamente mantienen la misma proporción. Lo mismo para las clases 11 y 12. Por lo que tendremos un total de 7 clases.

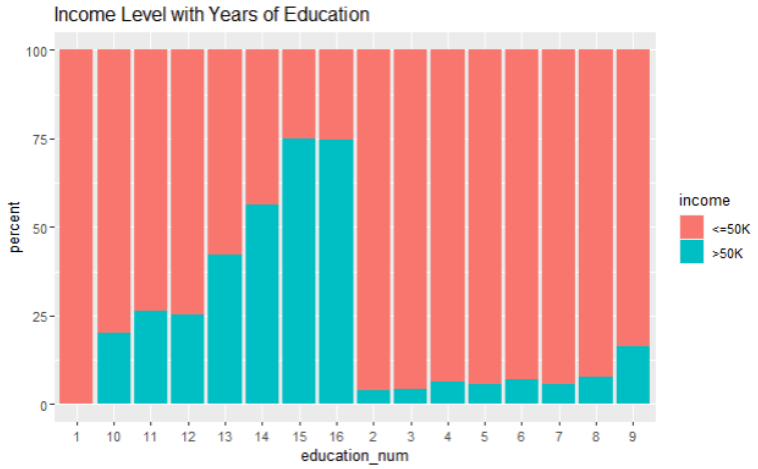

Figura 2: Histograma de la proporción entre los ***income*** para cada clase de ***education-num***

Para no llenar el trabajo entero de gráficos sobre las distribuciones de las variables, me limitaré a comentar los cambios que haré y los gráficos se encontrarán en los Anexos. Entonces seguiré con la variable ***occupation***, la dividiremos en 6 clases:

1. *Priv-house-serv*, *Other-service* y *Handlers-cleaners*
2. *Armed-Forces*, *Farming-fishing*, *Machine-op-inspct* y *Adm-clerical*
3. *Sales*
4. *Transport-moving* y *Craft-repair*
5. *Tech-support* y *Protective-serv*
6. *Prof-specialty* y *Exec-managerial*

En general, los grupos no tienen mucho en común así que los numeraremos simplemente del 1 al 6.

En cuanto al ***marital-status*** podemos dividirlo en 2 clases según la distribución, si están o no casados actualmente, teniendo en cuenta que los de la clase *Married-spouse-absent* cuenta como no casado.

Por lo que se refiere a ***race***, no haremos ningún ajuste, ya cuenta con 5 clases, cosa que parece razonable. Además puede ser interesante conservar estas clases tal y como están, pues posiblemente prescindamos de ***native-country***. Lo más probable es que nos den una información muy parecida.

La variable ***relationship*** puede recordar bastante a ***marital-status***, pero antes deberemos analizar su distribución de los grupos. Se pueden diferenciar claramente dos grupos, tal y como pasaba antes, uno de casados y otros de no casados. Por lo que no tendremos en cuenta esta variable. De hecho la única diferencia entre esta variable y ***marital-status*** es que esta es una combinación entre ***age*** y ***sex***, pero en ningún caso nos aporta con más información.

Por último, analizaremos a la variable ***native-country***. Es la que presenta un gráfico menos claro, pues hay una gran predominancia de *Estados Unidos* y hay muchas clases. Se podría dividir en 4 grandes grupos:

1. Paises con menos del 15 % que cobran más de 50k
2. Paises con menos porcentaje que *Estados Unidos* , pero más que un 15 %
3. *Estados Unidos*
4. Paises con más porcentaje que *Estados Unidos*

Si nos fijamos bien en que raza predomina en cada grupo de países vemos que hay bastante homogeneidad, por ejemplo en el grupo con % más bajo predominan países de raza blanca situados en Centro América o

Latino América, pero también hay países asiáticos. Por lo que realmente, sí que hará falta una combinación entre las variables ***native-country*** y ***race***.

Para terminar, estudiaremos el comportamiento de las variables numéricas del problema. La primera de ellas es ***age***, pero no hará falta hacer ningún tratamiento a estos datos. Podríamos dividirlo en intervalos, pero no ganaríamos nada. Simplemente perderíamos información.

En cuanto a las variables ***capital-gain*** y ***capital-loss*** podemos dividirlas en variables categóricas binarias por dos motivos:

1. El primero, y más evidente al ver la gráfica, es que prácticamente siempre que el valor de ***capital-gain*** es mayor que cero, entonces el ***income*** es mayor que 50.000 $ anuales. En el caso de ***capital-loss*** no es tan extremo, pero también pasa algo parecido.
2. También podemos ver que hay una cota superior de 99.999 para el valor de ***capital-gain***, por lo que podemos entender que no hace falta saber cuanto capital se ha ganado, si no si el balance será mayor que cero o no.

Por ese motivo nos planteamos crear una variable conjunta que se llame ***capital-aum***, con valor de cero si el capital no ha aumentado y con valor de 1 si lo ha hecho.

La última variable que tendremos en cuenta será la de ***hour-per-week***, la cual no modificaremos por los mismos motivos que con la edad. Además se mueven por los mismos rangos por lo que prácticamente no será necesario normalizar los datos, aunque lo haremos para comprobar si realmente se mejora o no.

Si quisiéramos hacer el análisis más preciso tendríamos que tener en cuenta la variable ***fnlwgt***, pues indica el peso y la representación que tiene cada fila. Pero no quería pecar de sobreajustamiento, ya que entonces las filas con más peso, cogerían demasiada importancia en el análisis y las filas menos representativas serian prácticamente invisibles. Por lo que, cuando apliquemos los algoritmos tampoco tendremos en cuenta esta variable.

Lo que acabamos de hacer también se conoce como *blocking*, que básicamente es disminuir el número de clases que hay para cada variable. De esta forma simplificaremos los modelos que obtendremos luego, ya que para las variables categóricas, algunos algoritmos los que hacen es considerar a cada grupo como una nueva variable. Evidentemente esto implica una pérdida de información, pero no toda la información es buena, así también podemos hacer que posibles outliers pertenecientes a los grupos minoritarios no influyan tanto en el modelo.

3. Algoritmos

Una vez hecho el preprocesado, ahora desarrollaremos distintos métodos que hemos visto en clase para solucionar este problema. Todos los modelos partirán de los mismos datos y usarán las mismas variables. Lo que hemos hecho ha sido crear una partición entre los datos de test y de training, un 80 % serán de training y el 20 % restante de test.

3.1. Regresión logística

La regresión logística sirve para modelar la probabilidad de un evento en función de otros factores, que en nuestro caso serán las variables que ya hemos ido estudiando. Este modelo se engloba en el conjunto de los *GLM* (Modelos lineales Generalizados) y usa la función logit:

$$logit(p) = log(\frac{p}{1-p}) = log(p) - log(1-p) \tag{1}$$

El modelo se puede expresar como lo que aportan unos datos(Y) y las variables explicativas(X) sobre la probabilidad final de pertenecer a un grupo o a otro y el modelo entonces toma la forma:

$$p_i = E(\frac{Y_i}{n_i}|X_i) \tag{2}$$

Los logits de las probabilidades son modelados como una función lineal de las variables explicativas:

$$logit(p_i) = ln(\frac{p_i}{1-p_i} = \beta_0 + \beta_1 x_{1,i} + ... + \beta_k x_{k,i} \tag{3}$$

Finalmente, el modelo se formula de la siguiente manera:

$$p_i = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_{1,i} + ... + \beta_k x_{k,i})}} \tag{4}$$

Como ya hemos venido haciendo, todos los códigos los escribiremos en R. Ejecutando la Regresión Logística podemos obtener fácilmente los coeficientes β de los que estábamos hablando. Podemos ver que algunos de ellos prácticamente son insignificantes y si quitáramos esas variables del modelo, prácticamente seguiría siendo el mismo. Por ejemplo, el coeficiente asociado a *workclassWithout-pay*, el cual solo tiene tres representaciones en nuestro conjunto de datos de training y todas ellas ganan menos de 50K, por eso motivo también cuenta con el pendiente más negativo, pues si un dato pertenece a este grupo las probabilidades que gane más de 50K són prácticamente nulas. Otro factor con pendiente bastante negativo es el *maritall-statusNo-casado*, de nuevo esto informa que alguien no casado tiende a cobrar menos de 50K al año. El que parece ser el componente más importante para decidir la probabilidad de cobrar más de 50K es la variable ***education***, pues hay una diferencia bastante grande entre las diferentes clases.

```
Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)                -4.09956    0.29731 -13.789  < 2e-16 ***
age                         0.37517    0.02224  16.868  < 2e-16 ***
workclassPrivate            0.11233    0.05598   2.006  0.04481 *
workclassSelf-emp-inc       0.32364    0.10217   3.168  0.00154 **
workclassSelf-emp-not-inc  -0.41582    0.08045  -5.169 2.36e-07 ***
workclassWithout-pay      -12.19473  137.83751  -0.088  0.92950
education_num2              0.94410    0.09014  10.474  < 2e-16 ***
education_num3              1.26591    0.09443  13.406  < 2e-16 ***
education_num4              1.45101    0.10660  13.612  < 2e-16 ***
education_num5              2.08796    0.09751  21.412  < 2e-16 ***
education_num6              2.40765    0.11581  20.790  < 2e-16 ***
education_num7              2.95000    0.13965  21.125  < 2e-16 ***
marital_statusNo-Casado    -2.32610    0.05111 -45.513  < 2e-16 ***
occupation2                 0.56727    0.09736   5.826 5.66e-09 ***
occupation3                 1.11827    0.10220  10.942  < 2e-16 ***
occupation4                 0.87197    0.09524   9.156  < 2e-16 ***
occupation5                 1.41496    0.11446  12.362  < 2e-16 ***
occupation6                 1.53604    0.09515  16.144  < 2e-16 ***
raceAsian-Pac-Islander      0.17572    0.27525   0.638  0.52321
raceBlack                   0.42199    0.25134   1.679  0.09317 .
raceOther                  -0.35408    0.41899  -0.845  0.39807
raceWhite                   0.44416    0.24033   1.848  0.06458 .
sexMale                     0.10891    0.05535   1.968  0.04910 *
hours_per_week              0.32746    0.02117  15.469  < 2e-16 ***
native_country2             0.22176    0.28231   0.786  0.43216
native_country3             0.59599    0.13265   4.493 7.03e-06 ***
native_country4             0.76420    0.17060   4.480 7.48e-06 ***
capital_aum                 1.56991    0.06254  25.101  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figura 3: Valor de los coeficientes β

Una vez obtenido el modelo, tendremos que hacer las predicciones para el particionado de test y comprobar como encaja el modelo. El resultado es la siguiente tabla:

pred	<=50K	>50K
<=50K	4138	642
>50K	362	890

Figura 4: Confusion Matrix generada por el modelo de la Regresión Logística

En la diagonal principal se encuentran los elementos que se han clasificado correctamente y en la otra diagonal se encuentran los falsos positivos y falsos negativos, que serán los datos que se han clasificado de forma errónea. Esto nos permite calcular la precisión de clasificación de este algoritmo, que será del 83.36 %. Lo que implica un error del 16.64 %. Si nos fijamos de nuevo en la Confusion Matrix, podemos ver que tenemos un modelo sobreajustado para una de las dos clases. Eso es debido a que habían muchos más datos que cobraban menos de 50K y por lo tanto el modelo tiende a clasificar hacia la clase que cobra menos de 50K, es por eso que el error de la otra clase es significativamente mayor. Una posible solución a esto podría haber sido escalar los datos para tener el mismo número de las dos clases, pero he considerado oportuno no hacerlo pues estaríamos perdiendo mucha información importante.

3.2. Support Vector Machine

Es uno de los algoritmos más utilizados para la clasificación en Aprendizaje Automático, lo que intenta hacer este algoritmo es separar espacialmente las dos clases por un hiperplano, de tal forma que la distancia entre las dos clases sea lo más amplia posible. Es por esto que se trabaja con dos hiperplanos paralelos que delimitan a las clases y estos se conocen como *Support Vectors*. Este algoritmo solo sirve para datos separables linealmente pues literalmente estamos construyendo un hiperplano en nuestro espacio n-dimensional que nos separa los datos en 2. Para trabajar con conjuntos de datos no separables linealmente se utilizan funciones de kernel. El algoritmo corresponde al siguiente problema de optimización:

$$\begin{aligned} \min_{(w,\gamma,s)\in\mathcal{R}^{N+1+m}} \quad & \tfrac{1}{2}w^T w + \nu\sum_{i=1}^{m} s_i \\ \text{s. to} \quad & y_i(w^T\phi(x_i)+\nu) + s_i \geq 1, \quad i = 1, ..., m \\ & s_i \geq 0 \end{aligned}$$

Consideraremos en nuestro caso la ν por defecto que será de 0.2, también conocida como *slack*. Esto permite ajustar mejor para modelos que no son perfectamente separables y lo más probable es que nos encontremos en este caso pues en los problemas lineales prácticamente nunca los datos son tan homogéneos y fáciles de clasificar. De momento no usaremos ningún kernel. Los resultados que obtenemos son los siguientes:

m3.pred	<=50K	>50K
<=50K	4160	642
>50K	340	890

Ahora tenemos una precisión de test del 83.74 % y un error del 16.26 %. De nuevo nos encontramos en la misma situación que antes, el modelo se ajusta muy bien para la clase mayoritaria pero muy mal para la minoritaria. Por lo que nos puede ser conveniente crear un apartado después de analizar los algoritmos para estudiar que pasaría si las clases estuvieran balanceadas.

3.3. Random Forest

Este será el último algoritmo que analizaremos, se trata del Random Forest. Esta técnica se basa en el *Bagging*, que quiere decir promediar muchos modelos ruidosos, para quedarse con lo que tienen en común, de tal forma disminuir la variación. La gran diferencia con otros algoritmos que usan *bagging* es que está compuesto por árboles predictores. Este algoritmo determinará la clase de un dato utilizando la clase que la mayoría de árboles haya predicho. Nosotros consideraremos 1000 árboles distintos.

Los resultados de aplicar este método son los siguientes:

Evidentemente el desbalance entre clases sigue presente, pero parece que este método ha sobrepasado a los

m4.pred	<=50K	>50K
<=50K	4143	593
>50K	357	939

otros dos. Esto lo comprobaremos calculando el porcentaje de acierto y de error, que son 84.25 % y 15.75 %. De momento el más bajo hasta ahora.

4. Balanceo de clases

Por último, comprobaré que hubiera pasado si las clases estuvieran balanceadas en los datos de traning. Mi predicción es que el porcentaje de acierto no cambiará demasiado. Si bien es cierto que de esta forma la clase que cobra más de 50K tendrá una mejor clasificación, la otra, que seguirá siendo mayoritaria en el test será mucho peor, pues habremos perdido información.

Por lo tanto lo que haremos será quedarnos con 5.976 elementos de cada clase para el set de training y coger 3189 elementos aleatorios de la clase test.

Empezamos con la Regresón logit y ahora tenemos un 19.7 % de error:

pred	<=50K	>50K
<=50K	1845	119
>50K	509	716

En el SVM tendremos un error de 20.88 %:

m6.pred	<=50K	>50K
<=50K	1795	107
>50K	559	728

Y por ultimo, el Random Forest con un error de 20.45 %:

m7.pred	<=50K	>50K
<=50K	1816	114
>50K	538	721

Como podemos ver, en todos los casos empeoramos los resultados. Tal y como habíamos predicho, las dos clases ahora se estiman con un error muy parecido, pero la clase mayoritaria sale muy mal parada en comparación. Para solucionar esto podríamos plantearnos hacer un estudio con pesos en las clases, para de esta forma compensar la proporción entre clases y a la vez no eliminar información.

5. Conclusiones

En conclusión, hemos determinado que el mejor modelo ha sido el generado a través del algoritmo Random Forest con un error del 15.75 %. Aunque hace falta comentar que no es excesivamente mejor, pues la diferencia con los otros algoritmos no llega ni al 1 %.

Uno de los principales problemas del Random Forest es la dificultad para interpretar sus modelos, por lo que me basaré en el modelo de Regresión Logística para entender que importa realmente a la hora de determinar si alguien cobrará más de 50K anuales. Como ya hemos comentado, las variables más influyentes son ***education***, ***marital-status***, **occupation** y ***capital-aum***. Esto tiene bastante sentido con lo que esperaríamos, a excepción de ***marital-status***, al menos yo no era consciente de que la gente casada tendía a tener más dinero, pero realmente tiene sentido. Casarse en nuestra sociedad está relacionado con crear una familia, y para ello debes estar seguro de que la podrás mantener. Se podría pensar de otra forma, cuando la gente se casa, deja de socializar tanto y centran su vida en el trabajo y la familia. Al estar centrados en el trabajo, tienden a trabajar mejor y les recompensan por ello con aumentos salariales o ascensos. Estas son las únicas explicaciones que se me vienen a la cabeza, pero tienen bastante sentido.

Una de las principales limitaciones del trabajo ha sido no utilizar la variable ***fnlwgt***, es decir los pesos. Lo más fácil de implementar seria replicar los datos tantas veces como ponga en la variable del peso. Pero eso no parece muy viable pues llegaríamos a tener más de 6.000.000.0000 de entradas. Una alternativa a esto seria normalizar el valor de la variable, dividiéndola entre el valor más pequñeo de esta y luego replicar los datos tantas veces como este nuevo valor. Aun así, parece bastante inviable pues mi portátil no tiene grandes prestaciones ni de RAM ni de espacio en el disco, por ejemplo no podía hacer un Random Forest de 10.000 árboles, por lo que he hecho bien en no tener en cuenta esta variable. Aunque seguramente los resultados serian mejores para analizar como influyen las distintas variables al ***income***.

Esta práctica ha sido muy útil para adentrarse en el mundo del aprendizaje automático y como ya había mencionado en la introducción darme cuenta de los innumerables problemas que se pueden resolver con estas técnicas. Ahora estoy motivado para adentrarme en este campo y afrontar los problemas que había planteado en la introducción.

6. Anexos

6.1. Descripción de variables

- ***age***: variable numérica que corresponde a la edad, está definida en el intervalo de 17 a 90 años.

- ***workclass***: variable categórica que expresa el tipo de empleo, con las siguientes 8 clases:

    * *Federal-gov*: funcionario del gobierno federal.

    * *Local-gov*: funcionario del gobierno local.

    * *Never-worked*: nunca ha trabajado.

    * *Private*: trabajando para una empresa privada.

    * *Self-emp-inc*: empresario a cargo de una gran empresa.

    * *Self-emp-not-inc*: autónomo.

    * *State-gov*: funcionario del gobierno estatal.

    * *Without-pay*: en paro.

- ***fnlwgt***: variable numérica continua que da una aproximación al peso de una persona en la sociedad, eso quiere decir una aproximación del número de personas que representa cada fila. Toma valores entre 12.285 y 1.484.705.

- ***education***: variable categórica que expresa la cantidad de estudios. Tiene 16 clases, cada una presenta niveles de educación disintos.

- ***education-num***: variable numérica que representa la anterior variables, cada número representa una clase distinta y toma valores en el intervalo del 1 al 16. Donde 16 es el máximo número de estudios que se puede tener.

- ***maritial-status***: variable categórica que expresa el estado civil, puede tomar un valor de los siguientes:

    * *Divorced:* está divosciada.

    * *Never-married:* nunca se ha casado.

    * *Married-AF-spouse:* está casada con alguien de las fuerzas armadas.

    * *Married-civ-spouse:* está casada con un civil.

    * *Married-spouse-absent:* está casada, pero en el registro no está su pareja.

    * *Separated:* está separada.

    * *Widowed:* está viuda.

- ***occupation***: variable categórica que indica el tipo de profesión, puede tomar 14 clases distintas.

- ***relationship***: variable categórica que indica su función en la familia. Estas funciones pueden ser:
    * *Husband*: esposo
    * *Not-in-family*: no pertenece a una familia
    * *Other-relative*: otro tipo de familiar.
    * *Own-child*: hijo.
    * *Unmarried*: soltero.
    * *Wife*: esposa.
- ***race***: variable categórica que describe la raza. Puede pertenecer a una de las siguientes clases:
    * *Amer-Indian-Eskimo*: indio americano o esquimal.
    * *Asian-Pac-Islander*: asiático o de las islas del Pácifico.
    * *Black*: negro.
    * *Other*: otros.
    * *White*: blanco.
- ***sex***: variable categórica binária que indica el sexo, solo contempla *Female* (mujer) o *Male* (hombre).
- ***capital-gain***: variable numérica que indica la ganancia del capital, comprende valores entre 0 y 99.999.
- ***capital-loss***: variable numérica que indica la perdida del capital, comprende valores entre 0 y 4.356.
- ***hours-per-week***: variable numérica que corresponde a las horas de trabajo semanales, toma valores entre 1 y 99.
- ***native-country***: variable categórica que indica al país de nacimiento. Hay 41 clases, una por país.
- ***income***: variable categórica que queremos predecir, indica si se cobra más de 50.000 $ o no.

6.2. Gráficos de la distribucions de las clases de las variables

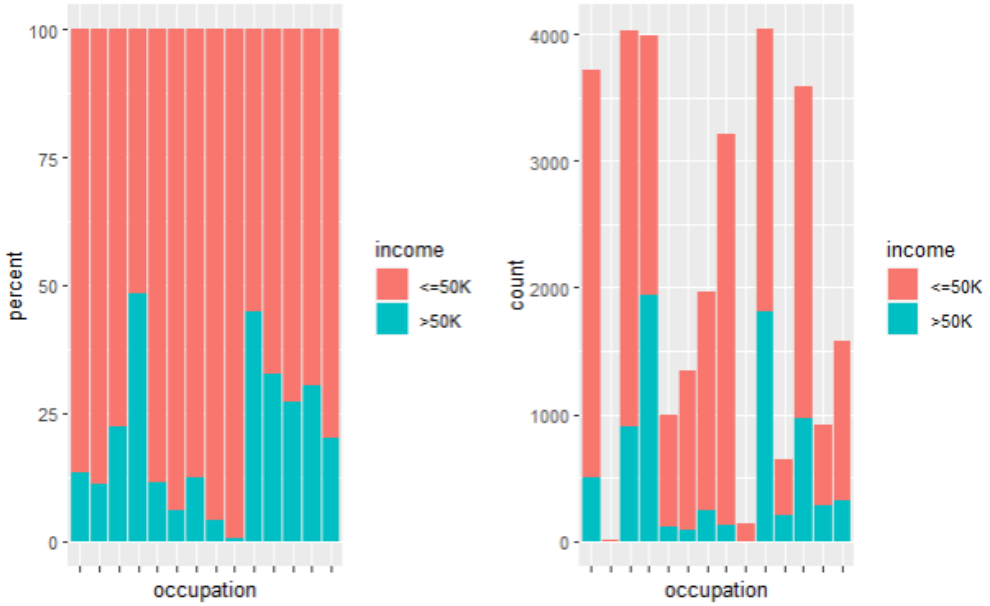

Figura 5: Histograma de la proporción entre los ***income*** para cada clase de ***occupation***

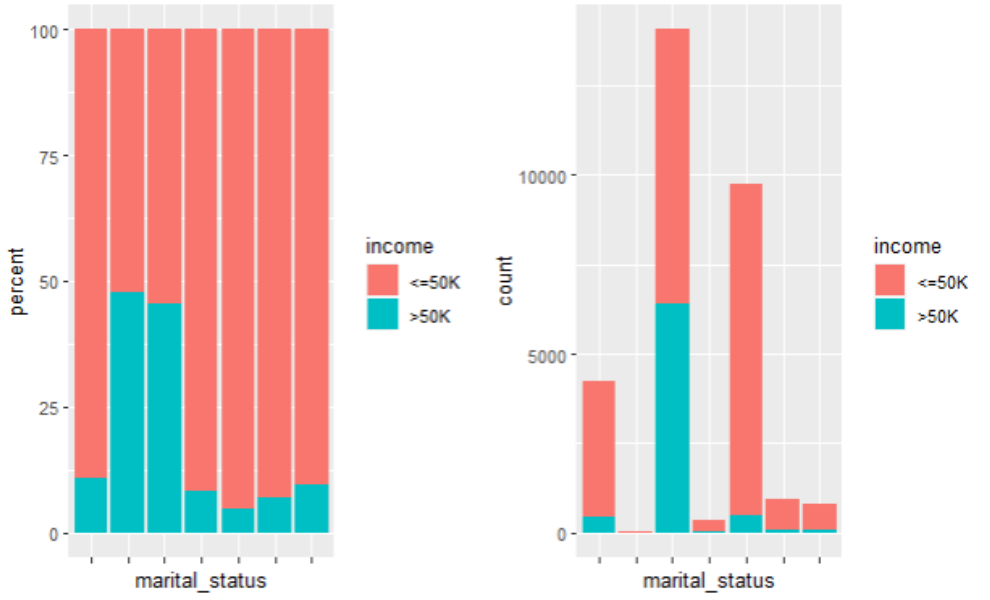

Figura 6: Histograma de la proporción entre los ***income*** para cada clase de ***marital-status***

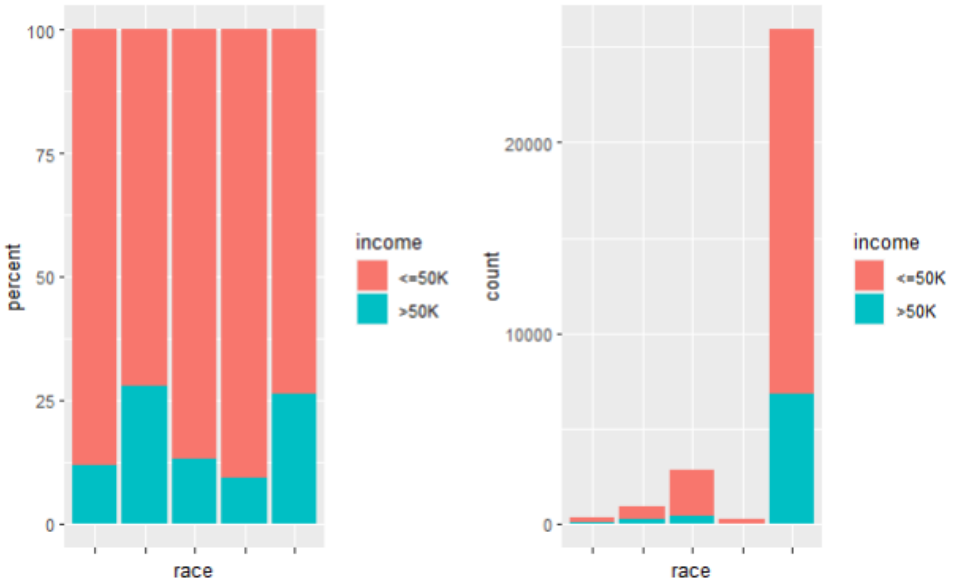

Figura 7: Histograma de la proporción entre los ***income*** para cada clase de ***race***

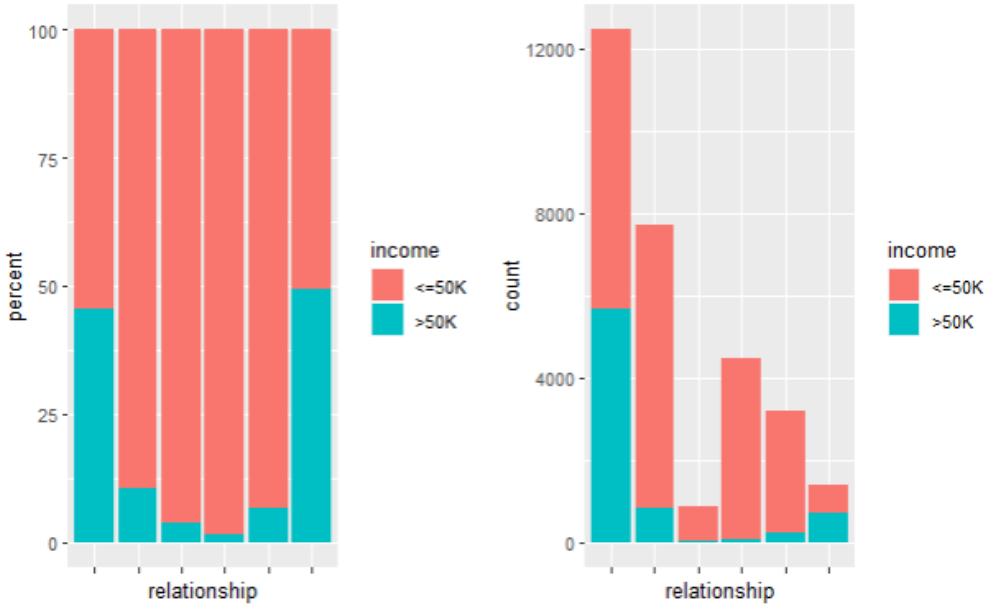

Figura 8: Histograma de la proporción entre los ***income*** para cada clase de ***relationship***

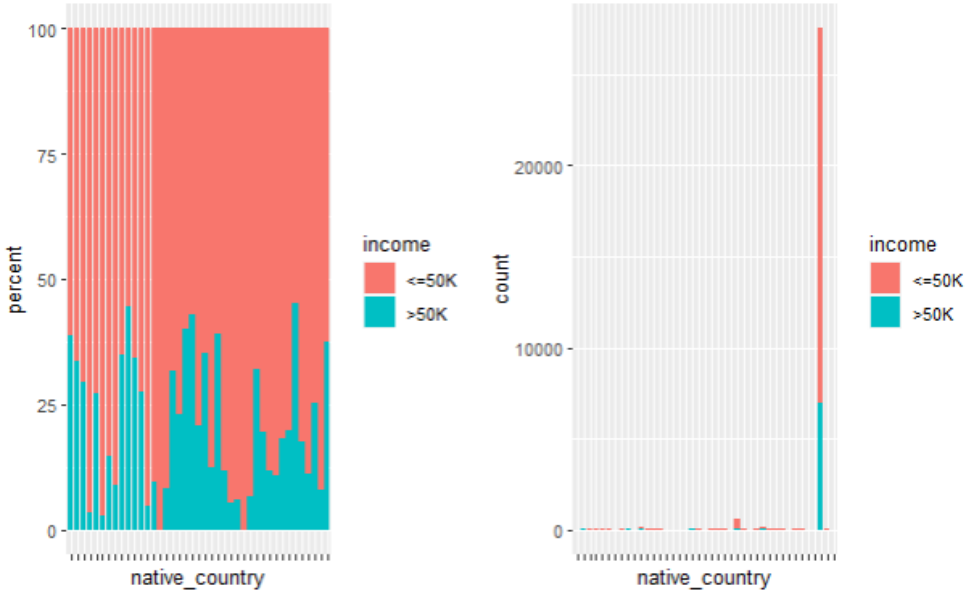

Figura 9: Histograma de la proporción entre los ***income*** para cada clase de ***native-country***