GRANDES DATOS, GOOGLE Y DESEMPLEO

BIG DATA, GOOGLE AND UNEMPLOYMENT

Raymundo M. Campos Vázquez

Sergio E. López-Araiza B.

El Colegio de México, A. C.

Resumen: Utilizamos datos de búsquedas en *Google* sobre empleo para pronosticar la tasa de desempleo en México. Discutimos la bibliografía relacionada con *nowcasting* y *big data* donde se utilizan datos generados en *internet* para predecir desempleo. Además, explicamos algoritmos de aprendizaje que sirven para escoger el mejor modelo de predicción. Finalmente, se aplican estos algoritmos para encontrar el modelo que mejor prediga la tasa de desempleo en México. En términos de políticas públicas, creemos que los datos generados a través de *internet* y los nuevos métodos estadísticos son claves para mejorar el diseño y la pertinencia de las intervenciones.

Abstract: We use Google Trends data for employment opportunities related reply in order to forecast the unemployment rate in Mexico. We begin by discussing the literature related to big data and nowcasting in which user generated data is used to forecast unemployment. Afterwards, we explain the basics of several machine learning algorithms. Finally, we implement such algorithms in order to find the best model to predict unemployment using both Google Trends queries and unemployment lags. From a public policy perspective, we believe that both user generated data and new statistical methods may provide great tools for the design of policy interventions.

Clasificación JEL/JEL Classification: C52, C53, E24, J64, O54

Palabras clave/keywords: desempleo; Google; grandes datos; aprendizaje automático; predicción; México; unemployment; Google; big data; machine learning; prediction

Fecha de recepción: 07 XI 2018 *Fecha de aceptación: 05 II 2019*

Estudios Económicos, vol. 35, núm. 1, enero-junio 2020, páginas 125-151

1. Introducción

Los estudios empíricos adquieren cada vez mayor importancia en la literatura de la ciencia económica (Einav y Levin, 2014a, 2014b). En particular, se ha visto una importante disminución de los artículos netamente teóricos frente a aquellos en los que se utilizan los datos para validar las hipótesis presentadas (Hamermesh, 2013). Al mismo tiempo, por avances tecnológicos y de poder computacional, los datos generados en internet han ido ganando relevancia para investigación científica. Esto ha generado un creciente interés desde la economía por conocer y aplicar los algoritmos de aprendizaje de máquina (*machine learning*) y grandes datos (*big data*) para responder las preguntas relevantes de la disciplina.

El presente artículo se suma a la nueva corriente de literatura que ha comenzado a utilizar los métodos de aprendizaje de máquina para realizar investigación científica aplicada. En concreto, presentamos un entorno muy específico en el que tanto estos nuevos modelos como las nuevas fuentes de datos que el internet ha traído consigo son de gran utilidad para la ciencia económica en México.

Nuestra intención es, por un lado, mostrar cómo es que los métodos de aprendizaje de máquina están siendo utilizados en la literatura económica a nivel internacional y, por otro, demostrar su utilidad para estudiar la economía mexicana. Además, queremos introducir algunos de estos métodos y cómo es que son instrumentados en la computadora para mostrar al gremio de economistas mexicanos cómo es que pueden ser utilizados en otras agendas de investigación.

Por otro lado, internet se ha convertido, en sí mismo, en una inmensa nueva fuente de datos. Las interacciones de los usuarios en redes sociales, las búsquedas personales, las ofertas de empleo y hasta las reseñas son potencialmente un insumo poderoso para contestar preguntas tradicionales (y no tradicionales) utilizando nuevas bases de datos. Este artículo, por lo tanto, también contribuye a diseminar dicha información para la academia mexicana.

Nuestra principal aportación consiste en mostrar cómo es que métodos propios del aprendizaje de máquina -como el LASSO (*Least Absolute Shrinkage and Selection Operator*) y los bosques aleatorios (*Random Forest*)-, aunados a datos de búsquedas en *Google*, pueden ayudarnos a predecir el valor de algunas variables macroeconómicas en el presente antes de que contemos con datos oficiales al respecto. Concretamente, buscamos predecir con la mayor precisión posible la tasa de desempleo nacional antes de que el Instituto Nacional de Estadística y Geografía (INEGI) publique la información oficial. Este

ejercicio no se ha realizado para la economía mexicana y representa una contribución importante en la bibliografía del tema.

Pese a que dichos métodos han demostrado ser particularmente eficientes para explorar las diferencias en efectos para subconjuntos heterogéneos de la población, este artículo pretende, únicamente, demostrar una aplicación sencilla de dichos algoritmos. En este sentido, si la capacidad predictiva que pueda tener el índice de *Google Trends*, utilizado tanto en modelos econométricos como en métodos de aprendizaje de máquina, mejora al analizar distintas subpoblaciones económicamente activas no será nuestro principal interés y, por este motivo, su análisis se limita a la fase exploratoria de los datos.

Tanto los nuevos métodos como los datos de *Google* traen consigo enormes ganancias predictivas. El método LASSO tiene un mejor desempeño que un modelo autorregresivo (conocido en la literatura como AR), incluso al incorporar datos de *Google*.

Por su parte, el bosque aleatorio tiene un desempeño apenas un poco inferior al LASSO. Finalmente, la precisión del modelo AR mejora de forma considerable cuando toma en cuenta datos de *Google*. Por tales motivos, argumentamos que la ciencia económica en México tiene que conocer y comenzar a utilizar estos métodos y nuevas fuentes de datos.

El resto del artículo está organizado como sigue. En la sección dos presentamos la abundante literatura que ha surgido de utilizar métodos de aprendizaje de máquina o datos de internet para contestar preguntas que son importantes en economía. Estos métodos han sido explotados desde muy diversos puntos de vista y con propósitos muy variados, por lo que conocer todo su potencial nos parece de enorme relevancia. Las secciones tres y cuatro están enfocadas en comprender mejor los datos que utilizaremos. En la primera, ofrecemos una breve explicación acerca de cómo se generan los datos de *Google* que utilizamos y exploramos sus principales características, mientras que en la segunda analizamos la relación que presentan estos datos con la Encuesta Nacional de Ocupación y Empleo (ENOE) que presenta el INEGI. En la sección cinco se discute la metodología que utilizaremos para construir y analizar los modelos y sus resultados. Explicamos en profundidad, particularmente, cómo evaluar de forma comparable el desempeño de los modelos, así como las implicaciones teóricas y prácticas del LASSO y del bosque aleatorio. La sección seis contiene el análisis de los resultados y en la sección siete presentamos nuestras conclusiones.

2. Bibliografía relevante

La ciencia económica ha visto en las últimas décadas un fuerte aumento de trabajos empíricos (Einav y Levin, 2014b). En este contexto, es natural que el uso tanto de fuentes de datos alternativas como de nuevos métodos de análisis estadístico estén incorporándose cada vez con mayor frecuencia a la literatura económica. Es cierto que el enfoque de dichas técnicas consiste en buscar mejorar las predicciones de los modelos y no, como en los métodos econométricos tradicionales, lograr identificar una relación causal entre dos variables (Einav y Levin, 2014b).

Sin embargo, su capacidad predictiva y las nuevas fuentes de datos han generado importantes ejemplos de su enorme valor para guiar políticas públicas (Athey, 2017; Banco Mundial, 2014) y para mejorar la investigación económica cuantitativa en general (Taddy, 2018). De hecho, estas nuevas técnicas estadísticas ya han comenzado a ser empleadas para responder a preguntas clave de las ciencias sociales, como mejorar la precisión de los pronósticos electorales alrededor del mundo (Gerunov, 2014; Kennedy, Lazer y Wojcik, 2017). Por otro lado, también se ha enfatizado la utilidad de dichos algoritmos para hacer que la toma de decisiones esté sistematizada y sea eficiente. Ejemplos de ello podemos encontrarlos en Erel *et al.* (2018) -en el ámbito de la conformación de los directivos de grandes empresas- o en Mullainathan y Obermeyer (2017), referente a diagnósticos médicos.

Otros usos de estos métodos para ciencias sociales abarcan diversos ámbitos del sector público, desde la educación hasta los departamentos policiacos. En el ámbito educativo, diversos estudios se han enfocado en predecir a los estudiantes de educación media con una mayor propensión a no terminar sus estudios, canalizando así de forma mucho más eficiente los recursos para apoyarlos (Sara *et al.* 2015; Lakkaraju *et al.* 2015; Aguiar *et al.* 2015). Por otro lado, estas herramientas han sido utilizadas también para identificar a los elementos policiales más propensos a tener un desencuentro con la sociedad (como, por ejemplo, hacer uso desmedido de la fuerza) y evitar así el desgaste de la imagen de la policía en Estados Unidos (Carton *et al.* 2016).

Se ha enfatizado también cómo el poder predictivo del llamado aprendizaje de máquina puede ser utilizado en el entorno de inferencia causal econométrico (Varian, 2014). En este sentido, se ha destacado su capacidad para encontrar la forma funcional adecuada (Athey, 2018), su probable uso para escoger el instrumento más adecuado en un estudio de variables instrumentales (Belloni, Chernozhukov y Hansen, 2014), contrastar empíricamente el desempeño de diversas

teorías (Mullainathan y Spiess, 2017) e identificar efectos heterogéneos en experimentos (Chernozhukov *et al.*, 2018).

En otro ámbito, de entre todas las fuentes de datos que el *Big Data* ha traído consigo, la más importante de ellas es el internet. Este tipo de datos tienen importantes ventajas, como su disponibilidad en tiempo real (Hilbert, 2016) y el alto volumen de observaciones que nos permite analizar datos con un nivel de granularidad que antes era impensable (Einav y Levin, 2014a). Además, los datos obtenidos a través de internet nos han permitido generar variables que antes eran inexistentes (Hilbert, 2016).

Una importante fuente de datos proveniente de internet para la ciencia económica han sido las redes sociales, que nos dan acceso tanto a las conversaciones y búsquedas como a los contactos y trayectorias individuales. Por tal motivo, las redes sociales han permitido a los economistas responder empíricamente preguntas que, hasta ahora, habrían sido imposibles de estudiar. En este sentido, Llorente *et al.* (2015), al utilizar las ubicaciones desde las cuales los españoles publicaban en la red social *Twitter*, generaron un modelo para predecir los datos de desempleo antes de que estos fueran publicados por el gobierno. De forma similar, el *Global Pulse Lab* (2014) de la ONU logró mejorar las predicciones de la inflación en Indonesia mediante las búsquedas en esta misma red social.

Otros estudios han logrado entender importantes procesos económicos utilizando datos de sitios electrónicos en los que se promueve algún tipo de intercambio económico entre los usuarios. Por ejemplo, Cavallo y Rigobon (2016) estimaron los niveles de inflación en Argentina entre 2007 y 2015 -una época en las que las estadísticas oficiales resultaban poco creíbles- utilizando datos de tiendas en línea. Asimismo, Einav *et al.* (2014) estudiaron cómo responden los consumidores a impuestos locales en sus compras por *Ebay*, mientras que Einav *et al.* (2018) analizaron el efecto de distintas estrategias de venta en la misma plataforma. Finalmente, Gleaser, Kim y Luca (2017) utilizaron datos de *Yelp*, una plataforma en la que los usuarios suben reseñas de diversos establecimientos comerciales, para predecir los niveles de actividad económica local en el presente (*nowcasting*).

El área de economía laboral también ha sido beneficiada por los datos provenientes de internet (Beblavy, Kurekova y Thum, 2014). Marinescu y Wolthoff (2016) utilizaron los datos de la página *careerbuilder*.com (plataforma en la que se demandan empleos y la gente puede postularse para obtenerlos) para entender cómo se emparejan las preferencias de la empresa con las de los posibles empleados, mientras que Marinescu (2017) los consideró para entender el efecto que

tienen las extensiones al seguro de desempleo en Estados Unidos sobre la búsqueda de empleo.

Otra importante fuente de datos para la literatura económica ha sido el índice de *Google Trends*, que muestra la intensidad de búsquedas en esta plataforma para determinados temas. Ha sido utilizado para hacer predicciones sobre el presente en torno a la venta de coches en Estados Unidos (Choi y Varian, 2012; Choi, 2010), las solicitudes de seguro de desempleo (Choi, 2010) y la demanda de viajes a determinados sitios turísticos (Choi y Varian, 2012). Además, *Google Trends* ha sido usado para medir el racismo a nivel estatal en Estados Unidos (Stephens-Davidowitz, 2017) o los efectos del seguro de desempleo en la búsqueda de empleo (Baker y Fradkin, 2017).

Un mayor número de ejemplos de predicciones del presente de la actividad económica con *Google Trends* lo proporcionan Tuhkuri (2015) y Goel *et al.* (2010). En el caso del primero, estima un modelo AR para predecir el presente de la tasa de desempleo a nivel estatal en Estados Unidos. De acuerdo con sus resultados, la información contenida en el índice de *Google Trends* mejora la capacidad predictiva de su modelo. Goel *et al.* (2010), por su parte, consideran el índice de *Google Trends* para predecir el comportamiento de los consumidores en diversos mercados: las ventas en taquilla en la semana de estreno de películas, ventas de videojuegos y el *ranking* en la lista de 100 canciones más escuchadas de *Billboard*. En todos los casos, observan que los índices de búsqueda pueden predecir bastante bien los resultados.

Esta explosión en nuevas fuentes de datos está íntimamente relacionada con una mayor capacidad para procesar y organizar bases cada vez más complejas. En este sentido, las técnicas propias del *Big Data* también son sumamente útiles en una etapa de procesamiento (Mullainathan y Spiess, 2017). Por ejemplo, Barjamovic *et. al* (2017) utilizaron algoritmos para procesar registros de transacciones entre comerciantes asirios del siglo 19 A.C. y, haciendo uso de modelos macroeconómicos de comercio, predecir en dónde podrían encontrarse las ruinas de antiguas ciudades de la Edad de Bronce.

Otro importante ejemplo de cómo las nuevas técnicas y el poder computacional con el que contamos actualmente está influyendo poderosamente en la ciencia económica lo proveen Bok *et al.* (2018). En un ejercicio similar al nuestro, presentan un modelo que les permite predecir en el presente, con un gran nivel de precisión, los cambios reales en el PIB de Estados Unidos utilizando índices publicados por diversas fuentes.

Finalmente, el uso de *Big Data* en ciencias sociales también ha sabido aprovechar las imágenes como fuentes de información. Rundle *et al.* (2011) utilizaron imágenes de *Google Street View* para codificar las condiciones de ambiente -seguridad peatonal y el tráfico motorizado de la zona, entre otras- de diversos barrios de Nueva York. Asimismo, se han utilizado imágenes satelitales nocturnas para generar medidas aproximadas de la riqueza de determinadas áreas basándose en su iluminación (Henderson, Storeygard y Weil, 2012).

Pese a que las técnicas y fuentes de datos propios del *Big Data* cada vez permean con mayor intensidad a la economía, es importante tener en mente múltiples consideraciones para poderlos utilizar e interpretar correctamente. Por un lado, es imprescindible tener en cuenta cuál es el proceso de generación de datos (Banco Mundial, 2014) que, a su vez, determina qué observaciones son observables y cuáles no. En este sentido, es importante tomar en cuenta que puede existir algún error de muestreo al tratarse de individuos que se autoseleccionan para aparecer como observables.

Para finalizar, existen también algunas complicaciones éticas y logísticas. Por un lado, los mayores recolectores de este tipo de datos son instituciones privadas. Lograr convenios de colaboración entre la academia con el sector privado para trabajar con dichos datos puede tornarse en una tarea complicada (Horton y Tambe, 2015). Por otro lado, la granularidad de los datos trae consigo un importante problema ético: garantizar la privacidad de las personas.

Este dilema es claro en una fuente de datos que cada vez es más común en el uso de *Big Data* para las ciencias sociales: las llamadas y localizaciones de los teléfonos celulares. Blumenstock y Donaldson (2013) analizaron patrones de llamadas desde teléfonos móviles en Ruanda para entender cómo la movilidad del factor trabajo pone en equilibrio los mercados laborales. Independientemente de que se trata de una gran aportación empírica para la economía, los protocolos para garantizar el anonimato de los individuos cuyas llamadas fueron rastradas es un tema de gran importancia.

En síntesis, el uso tanto de las nuevas fuentes de datos como de los nuevos métodos de análisis que conforman al llamado *Big Data* se ha insertado ya en la literatura económica con experiencias muy diversas. Sin embargo, pese a que el índice de *Google Trends* ha sido utilizado con herramientas econométricas tradicionales, no hemos encontrado aún una comparación entre los algoritmos de aprendizaje de máquina y dichos modelos econométricos cuando ambos incorporan la información de las búsquedas por internet. El presente estudio pretende mostrar cómo los nuevos métodos estadísticos y los datos de

internet pueden utilizarse para mejorar la predicción del presente en el caso mexicano. Esa será nuestra aportación a la literatura.

3. Tendencias en *Google*

En los últimos años, *Google Trends* se ha convertido en una poderosa herramienta para conocer los temas que más interesan a las personas en un momento en el tiempo. Al ser *Google* el motor de búsqueda más popular de nuestros días, la posibilidad de conocer cómo va evolucionando el interés en esta plataforma a lo largo del tiempo de determinados temas nos permite tener un pulso de cómo van moldeándose las preferencias y los intereses individuales. Además, las condiciones en las que los individuos suelen hacer búsquedas en esta plataforma digital -usualmente solos y desde el anonimato- hacen que dichos datos sean de gran utilidad para conocer pensamientos o preferencias que podrían ser socialmente cuestionadas (Stephens-Davidowitz, 2017). De este modo, podemos obtener un pulso en torno a temas sensibles, como el racismo o las preferencias electorales, que probablemente no reflejaría una encuesta tradicional debido a que los entrevistados tienden a dar la respuesta socialmente aceptable al entrevistador.

Google Trends nos permite obtener una serie de tiempo de un índice que representa el interés en determinados temas en la plataforma (Choi y Varian, 2012). Dicho índice es calculado con una muestra representativa de todos los datos de búsqueda de *Google* en el periodo de tiempo especificado. Pueden consultarse dos muestras distintas: en tiempo real o con datos del pasado. La primera está basada en una muestra aleatoria de los últimos siete días, mientras que la segunda se basa en otra muestra en un rango de tiempo que puede comenzar desde 2004 hasta 36 horas antes de la consulta (Rogers, 2016). Debido a que el ejercicio de muestreo se realiza con cada búsqueda que el usuario hace en *Google Trends*, los resultados pueden presentar diferencias mínimas, aun cuando se hagan dos búsquedas idénticas (Choi y Varian, 2012).

Sin embargo, la interpretación de la información que presenta *Google Trends* dista mucho de ser sencilla, pues lo que reporta es un índice que va de 0 a 100 en el periodo de tiempo seleccionado. De este modo, de acuerdo con la ayuda desplegada por el mismo *Google Trends*, "Los números reflejan el interés de búsqueda en relación con el mayor valor [...] en una región y en un periodo determinados. Un valor de 100 indica la popularidad máxima de un término, mientras que 50 y 0 indican una popularidad que es la mitad o inferior al 1%, respectivamente, en relación al mayor valor" (*Google*, s.f.).

Para calcular este índice, *Google* se basa en una medida de interés de los términos de búsqueda, y no en el total de búsquedas de dicho término en su motor. En este sentido, lo que *Google Trends* utiliza es el cociente del número de búsquedas del término especificado por el usuario sobre el total de búsquedas en *Google* en ese momento del tiempo. Por ejemplo, si le pidiéramos a *Trends* que nos desplegara información acerca del término de búsqueda "empleo", *Google Trends* utilizaría la ecuación 1 para calcular su índice:

$$interes = \frac{Busquedas\ empleo}{Total\ de\ busquedas\ Google} \tag{1}$$

Con estos cocientes, *Google Trends* calcula el índice normalizando a 100 el día para en el cual el término de interés obtuvo una mayor proporción de búsquedas y calcula el resto de los valores relativos al cociente de ese día.

Este modo de calcular el índice trae consigo ciertas ventajas. Al reflejar el interés en un tema, y no el total de búsquedas del mismo, la comparación entre diferentes regiones y distintos periodos tiene mucho más sentido: el número de búsquedas en el estado de Guerrero podría ser potencialmente mucho menor al de la Ciudad de México, para cualquier tema, simplemente por el número de habitantes y por la penetración de internet. No obstante, la medida de intensidad utilizada por *Google* permite que ambos sean comparables al reflejar un índice relativo al total de búsquedas en cada entidad. Lo mismo podría decirse a través del tiempo: en el año 2004 el número de búsquedas en *Google* de cualquier tema es probablemente menor al que observaríamos ahora, sencillamente porque *Google* no era tan popular en 2004 como lo es hoy.

Sin embargo, el uso de este cociente para calcular el índice presentado con *Google Trends* hace que sea un poco más complicado interpretar los cambios en la tendencia del índice a lo largo del tiempo. Esto se debe a que la disminución en el valor del índice de un día al otro, para un determinado término de búsqueda, puede explicarse tanto por un menor número de búsquedas asociadas a este término como por un aumento en el resto de las búsquedas de *Google*. Lo que puede traer consigo ventajas y desventajas, dependiendo del tipo de análisis que pretenda hacerse con los datos de *Google Trends*. Por tal motivo, cualquier análisis que utilice estos datos debe hacerse con mucha precaución, siempre tomando en cuenta que se presenta un índice basado en la proporción de búsquedas, y no en el total.

Otra ventaja de *Google Trends* es que pueden hacerse búsquedas por términos o por temas. Mientras que las primeras arrojan los resultados para los términos de búsqueda especificados por el usuario (y muestran resultados exclusivamente para el idioma en el que se especificaron), los segundos agrupan a todos los términos de búsqueda específicos a un tema, en cualquier lenguaje. Por ejemplo, la búsqueda del tema "Londres" arrojaría un índice basado en la búsqueda de los términos "Londres", "London" y "capital de Inglaterra", entre otros. Además, en el caso de las búsquedas por términos, el modo en el que está escrita la búsqueda (en términos de mayúsculas, minúsculas y acentuación) es importante: el resultado puede ser diferente para "Líbano" que para "libano" o "líbano".

Finalmente, hay que considerar también que *Google Trends* tiene ciertas limitaciones en términos de los datos que no muestra. Por un lado, no puede mostrar datos para búsquedas hechas por pocas personas: este tipo de búsquedas tendrá un índice con un valor de cero. Además, *Google Trends* excluye términos de búsqueda con caracteres especiales, como apóstrofes.

4. Correlaciones con la Encuesta Nacional de Ocupación y Empleo (ENOE)

Antes de describir los modelos a utilizar, conviene explorar primero cómo es que se relaciona la tasa de desempleo con el predictor que estamos proponiendo: el índice de búsquedas de *Google Trends* (IGT) para el término "empleo + 'bolsa de trabajo' ". Debido a que los otros predictores serán las tasas de desempleo rezagadas, esta es la única variable independiente relevante en la presente sección.

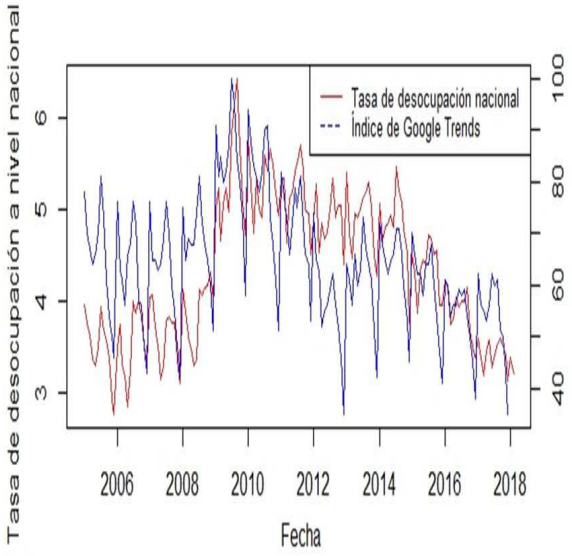
Creemos que puede existir una correlación importante entre ambas variables porque internet, cada vez con mayor presencia, es un medio importante de búsquedas de empleo en nuestro país. Además, de acuerdo con datos del Instituto Nacional de Estadística y Geografía (INEGI), el número de usuarios de internet en México va en aumento, incluso llegó a 60% de cobertura en 2016 (INEGI, 2017).

Por otro lado, como ya se mencionó anteriormente, existe evidencia de que los datos generados en internet, en general, (Gleaser, Kim y Luca, 2017; Bok *et al*, 2018) y las búsquedas de *Google*, en particular, son de gran utilidad para predecir el presente (Choi y Varian, 2012). Inclusive, el índice de *Google* ya ha sido utilizado con éxito para predecir el desempleo en Estados Unidos (Tuhkuri, 2015). Sin embargo, hasta donde tenemos conocimiento, aún no existe un

estudio que documente la utilidad de las búsquedas en *Google* para predecir las condiciones del mercado laboral en México.

En la gráfica 1, mostramos el comportamiento de la tasa de desempleo a nivel nacional tomada de la ENOE (eje del lado izquierdo) y del índice de *Google Trends* (eje del lado derecho):

Gráfica 1
IGT y tasa de desocupación a nivel nacional



Fuente: Elaboración propia con datos de la ENOE y del IGT para la búsqueda "empleo + 'bolsa de trabajo' ".

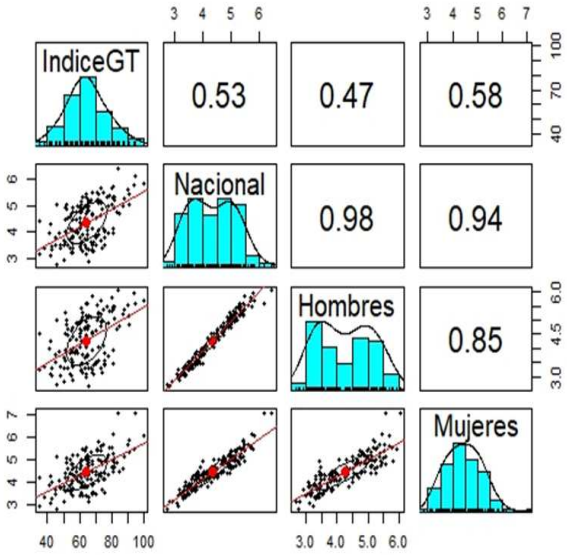
Es importante recordar que, pese a que ambos datos teóricamente fluctúan del 0 al 100, en realidad, debido a que tienen varianzas muy distintas, están en diferentes escalas. Como podemos observar, mientras que el IGT, efectivamente, toma valores que van del 40 al 100 a lo largo de los 13 años para los que disponemos de datos, la tasa de desocupación va de 3% a 6.4%. Por tal motivo, no resulta tan relevante que las líneas pasen exactamente por el mismo punto, sino que fluctúen de forma similar a través del tiempo.

En términos generales sí observamos, de hecho, un movimiento en la misma dirección de ambas curvas. Incluso en el periodo que va de 2008 a 2010, cuando la Gran Recesión estadunidense dejó estragos en la economía mexicana, el IGT registró movimientos muy

similares (aunque vale la pena recalcarlo, en otra escala) a la tasa de desocupación nacional. Por este motivo, el IGT sí parece ser un buen predictor para la tasa de desempleo nacional. Sin embargo, ¿se relaciona tan bien con la tasa de desocupación de hombres que con la de mujeres?

Pese a que encontrar este tipo de diferencias no es la motivación principal del presente artículo, creemos que podría ayudarnos a entender el perfil de usuarios de la plataforma de búsqueda más popular de internet. En la gráfica 2 se presenta una matriz de correlación (en número y en figura de dispersión) con el IGT y las tasas de desocupación de hombres, mujeres y promedio a nivel nacional. El primer renglón y columna se refieren al IGT, el segundo renglón y columna a la tasa de desocupación nacional y el tercer y cuarto renglón se refieren a la tasa para hombres y mujeres, respectivamente.

Gráfica 2
Matriz de correlaciones



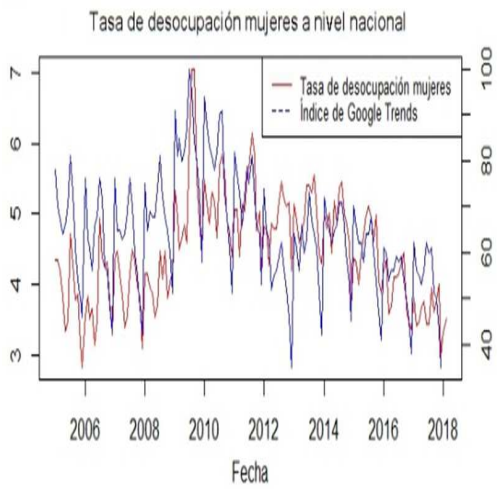
Fuente: Elaboración propia con datos de la ENOE y del IGT para la búsqueda "empleo + 'bolsa de trabajo' ". La gráfica muestra una matriz de correlaciones entre diversas variables. En la diagonal encontramos la distribución y el nombre de las variables, en la parte inferior izquierda los puntos de dispersión de las variables correlacionadas acompañado de una línea de tendencia entre ambas. Finalmente, la parte superior derecha presenta el índice de correlación de Pearson.

En primera instancia, vale la pena resaltar las correlaciones presentes en las tasas de desocupación. Claramente, como lo muestra la gráfica 2, la desocupación de los hombres está asociada de forma más cercana a la nacional (corr=0.98) que la de las mujeres (corr=0.94). Por otro lado, la correlación entre estas es mucho más débil: corr=0.85.

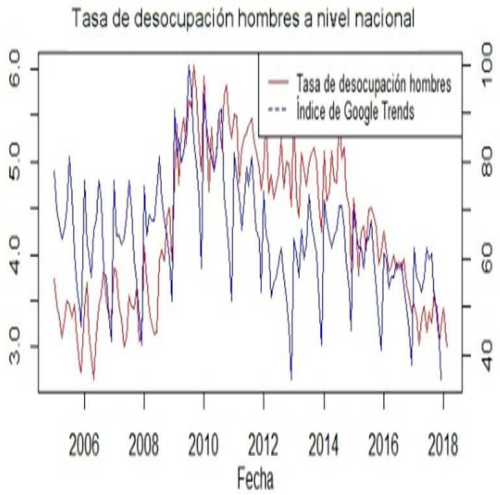
En lo referente al índice de *Google Trends* es claro que tiene una asociación más fuerte con la desocupación femenina nacional (corr=0.58) que con la masculina nacional (corr=0.47). No obstante, ambos valores están relativamente cercanos a la correlación que presentan con la tasa nacional (corr=0.53), por lo que podríamos esperar que, igual que como observamos en la gráfica anterior, el IGT y la desocupación nacional tanto de hombres como de mujeres se comportaran de forma similar a la nacional.

En efecto, como lo muestra la gráfica 3, para ambos casos el IGT y la tasa de desocupación se mueven en términos generales en la misma dirección. En síntesis, sí parece existir una relación importante entre el nivel de desempleo y el IGT para la búsqueda "empleo + 'bolsa de trabajo' ". En este sentido, usar los datos de *Google Trends* para predecir el presente del mercado laboral mexicano podría ser factible, siempre y cuando utilicemos los métodos adecuados. En la siguiente sección describiremos brevemente tres distintos métodos que usaremos para intentar predecir el desempleo nacional.

Gráfica 3
IGT y desocupación nacional por sexo



Gráfica 3
(*continuación*)



Fuente: Elaboración propia con datos de la ENOE y del IGT para la búsqueda "empleo + 'bolsa de trabajo' ".

Finalmente, podría resultar importante explorar cómo es que se correlaciona el índice de Google Trends con la tasa de desempleo para distintos subconjuntos de la población económicamente activa. El cuadro 1 muestra el índice de correlación de Pearson entre el índice de *Google Trends* y el desempleo de distintos subgrupos.

Cuadro 1
Correlación entre IGT y otras series de desempleo

Serie	*Índice de correlación de Pearson con el IGT*
Desempleo entre menores a 30 años	0.51
Desempleo entre quienes tienen entre 30 y 60 años	0.48
Desempleo entre los mayores a 60 años	0.31
Desempleo urbano	0.62
Hombres desempleo urbano	0.54
Mujeres desempleo urbano	0.66

Fuente: Elaboración propia con datos de la ENOE y del IGT para la búsqueda "empleo + 'bolsa de trabajo' ".

Debido a la frecuencia de estos datos -que es trimestral y no mensual, como las series que ocupamos para estimar los modelos-, aunado a las restricciones de espacio del presente artículo, explorar más a fondo dichas heterogeneidades cae fuera de los límites de este trabajo. Sin embargo, consideramos que hacerlo sería de suma importancia para futuras investigaciones.

5. Aprendizaje automático: metodología

El estudio que aquí presentamos pretende demostrar dos cosas. Por un lado, que los datos provenientes de fuentes electrónicas pueden ser de gran utilidad para predecir el presente (*nowcasting*) y, de este modo, obtener medidas económicas preliminares antes de que los datos oficiales estén disponibles, aun utilizando métodos econométricos tradicionales. Más allá de la ganancia predictiva de estas nuevas fuentes de datos, obtener información relativamente confiable del desempeño económico en tiempo real puede ser de gran utilidad para implementar a tiempo políticas públicas informadas y eficientes. Por otro lado, buscamos también acercar a la comunidad de ciencias sociales los nuevos métodos del aprendizaje de máquina y evaluar su capacidad predictiva *vis a vis* las técnicas econométricas populares en la literatura.

Se utilizarán los datos de desempleo de la ENOE así como el índice de *Google Trends* para la búsqueda "empleo + 'bolsa de trabajo' " tanto a nivel nacional como estatal. La variable independiente será la tasa de desempleo en el periodo t y construiremos nuestro modelo predictivo con dicha variable rezagada algunos periodos, así como con el índice de *Google Trends*.

5.1. *¿Cómo medimos los errores?*

Quizás la principal diferencia entre la econometría tradicional y el aprendizaje de máquina es que ambas están diseñadas para hacer cosas distintas: mientras que la econometría tradicional busca estimar los parámetros del modelo (y por eso buscan estimaciones insesgadas que se acerquen al "valor verdadero" del parámetro), los algoritmos de *machine learning* están enfocados en mejorar la capacidad predictiva del modelo (Mullainathan y Spiess, 2017; Athey, 2018). Lo que implica que ya no importa cuál es el valor "real" de los parámetros, sino que el valor que arroja el modelo sea el más útil para predecir nuevas observaciones.

Por este motivo, la medición del error del modelo en la literatura de aprendizaje de máquina se hace de forma distinta a los métodos econométricos. Mientras que en econometría buscamos reducir el error cuadrático medio (ECM) dentro de la muestra sobre la que se estima el modelo (Hayashi, 2011), en aprendizaje de máquina (*machine learning*) se utiliza el método de validación cruzada (*cross validation*): se dividen los datos en dos submuestras, una de entrenamiento (para definir el valor de los parámetros del modelo) y otra de validación, para medir el error predictivo del modelo (Hastie *et al.*, 2013; Varian, 2014).

Por ejemplo, asumamos que dividimos los datos en cinco partes iguales, de tal suerte que cada una de ellas contiene 20% de las observaciones (conocido en la literatura como *five-fold cross validation*). Utilizaríamos 80% de las observaciones (grupos 1 a 4) para entrenar el modelo y el 20% restante para validarlo. Luego repetimos el ejercicio, pero con grupos distintos (entrenamos con grupos 2 a 5 y validamos con grupo 1). Nos quedamos con el modelo que tenga el ECM fuera de la muestra más pequeño (Hastie *et al.*, 2013).

Dicho procedimiento tiene sentido cuando se trata de datos de sección cruzada. Sin embargo, en nuestro caso, lo que tenemos es un modelo en el que nuestras observaciones tienen un fuerte componente de correlación serial. Pese a que hay quien argumenta que puede utilizarse el mismo procedimiento para dividir a los conjuntos de entrenamiento y de validación, consideramos que es pertinente utilizar métodos que tomen en cuenta tanto la estructura de los datos como el objetivo de predecir a futuro el problema.

Por tal motivo optamos por utilizar un procedimiento de validación cruzada con ventana cambiante (*rolling window cross validation*, más detalles en Hotal, Handa y Shrivas, 2017). Funciona de la siguiente manera: se seleccionan n número de periodos (le llamaremos "ancho de la ventana") y se utilizan los primeros n datos para entrenar al modelo y, con él, predecir el valor de la variable en el siguiente periodo. Puesto que conocemos el valor de la variable en el periodo $n+1$, podemos medir la precisión del modelo al compararlo con el valor de la predicción.

Esto nos da una primera aproximación de validación cruzada. Posteriormente, avanzamos solo un periodo y se repite el procedimiento, de tal suerte que se utilizan los datos que van desde que $t = 2$ hasta $t = n + 1$ para predecir la observación en $t = n + 2$ y calcular el error de predicción. El procedimiento se repite hasta haber agotado los datos. La medida de error del método utilizado será el promedio de los ECM de cada estimación (Inouea, Jinb y Rossi, 2017).

Para contrastar y comparar diferentes técnicas de estimación, se utilizará dicha metodología para calcular los errores de todos los modelos. De este modo, tendremos una medida comparable entre los diferentes casos que nos podrá mostrar las fortalezas y debilidades de cada método.

En este sentido, dado que nuestro interés es comparar la capacidad predictiva de los modelos, vale la pena aclarar que los parámetros estimados pueden diferir a través del tiempo. Lo que se debe a que se estimará el modelo en cada "ventana", por lo que el valor de los estimadores dependerá de los datos que se encuentren en esta submuestra de nuestro conjunto de datos.

5.2. *Modelo autorregresivo*

La primera especificación empírica que utilizamos para predecir el nivel de desempleo en las entidades federativas proviene, con base en Choi y Varian (2012), de modelos macroeconométricos tradicionales. En relación con esto, utilizamos modelos AR con información de la ENOE para predecir los niveles de desocupación en los trimestres posteriores y analizamos si mejora la capacidad predictiva de dichos modelos al incorporar búsquedas de *Google Trends* asociadas con búsquedas de empleo.

En concreto, el procedimiento compara la precisión de dos diferentes modelos, descritos en las ecuaciones 2 y 3:[1]

$$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-12} + \epsilon_t \tag{2}$$

contra

$$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-12} + \gamma T_t + \epsilon_t \tag{3}$$

En donde IGT corresponde al índice de búsquedas de *Google Trends*.

[1] Esta es la versión más simple para un modelo autorregresivo, que nos pareció sensata para la estimación del modelo: el rezago de un mes nos ayuda a incorporar la tendencia temporal del desempleo, mientras que el rezago anual nos permita capturar la estacionalidad de la demanda laboral. Sin embargo, en el anexo B se pueden encontrar los resultados para estimaciones con modelos de AR con más rezagos. Como podrá verse, las diferencias de precisión son marginales.

5.3. *Otros modelos de análisis estadístico*

Por otro lado, queremos poner a prueba la capacidad predictiva de los algoritmos desarrollados en el ámbito del aprendizaje de máquina para variables de índole económica. Para hacerlo, probaremos con dos algoritmos distintos para generar predicciones: el método LASSO y un bosque aleatorio (*Random Forest*).

5.3.1. Método LASSO

El primero de ellos, el método LASSO, es quizás el algoritmo de aprendizaje de máquina más utilizado por los economistas. Esto se debe, principalmente, a su enorme parecido con la regresión lineal clásica (Varian, 2014). El problema de minimización al que se enfrenta el método LASSO es el que muestra la ecuación 4 (Hastie *et al.*, 2013):

$$\hat{\beta} = argmin \sum_{i} (Y_i - \beta'(X_i))^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{4}$$

Como resultado, LASSO determina que el valor de algunos parámetros de interés debe ser cero, ya que está penalizando aquellos que toman cualquier valor distinto (Hastie *et al.*, 2013). Además, como resulta evidente, el valor que tomen los parámetros es, al final, una función del valor de lambda. Por este motivo, es importante seleccionar el valor de λ que minimice el error cuadrático medio utilizando validación cruzada (Belloni, Chernozhukov y Hansen, 2014).

Aunque se tengan pocas variables independientes que resulten de interés, el método LASSO nos permite incorporar al análisis una buena cantidad de interacciones entre ellas. En este sentido, a diferencia de lo que sucedería con una regresión tradicional, el método nos permite explorar múltiples relaciones no lineales sabiendo que, en última instancia, buena parte de las interacciones que probemos tendrán un coeficiente de cero (Mullainathan y Spiess, 2017).

LASSO refleja de forma importante la principal diferencia entre la econometría y el aprendizaje de máquina de la que hablábamos al inicio de esta sección (estimación *vs.* predicción). Lo anterior se debe a que, puesto que el problema de minimización del método obliga a que muchos de los coeficientes tengan un valor de cero, el resto de los coeficientes (que serán los que mejor nos ayuden a predecir futuras

observaciones) tendrán un sesgo de variable omitida (Varian, 2014; Belloni, Chernozhukov y Hansen, 2014).

Para el problema de predicción utilizaremos el método LASSO para predecir el valor de la tasa de desocupación considerando valores rezagados tanto de esta variable (desde $t-1$ hasta $t-12$) como del IGT (desde t hasta $t-12$). Además, para explotar la flexibilidad del método, incorporaremos al análisis las interacciones cúbicas de las variables para explorar relaciones no lineales. De este modo, intentaremos predecir el modelo descrito en la ecuación 5:

$$u_t \sim \left(\sum_{i=t-12}^{t-1} u_i + \sum_{j=t-12}^{t} IGT_i \right)^3 \tag{5}$$

5.3.2. Bosque aleatorio (*Random Forest*)

El segundo método que utilizamos es un bosque aleatorio. Es uno de los métodos más populares para la comunidad de aprendizaje de máquina ya que tiene una gran capacidad predictiva sin la necesidad de afinar excesivamente los hiperparámetros (Varian, 2014).

Un bosque aleatorio está conformado, a su vez, por múltiples árboles de decisión. Cada árbol de decisión es un problema independiente[2] que busca minimizar el error cuadrático medio (ECM) de las predicciones partiendo el espacio de características en cada nodo. De este forma, selecciona la variable que, al ser dividida en dos regiones distintas, predice mejor los resultados y minimiza el ECM. Una vez terminado el árbol, a cada observación se le asigna como predicción el valor promedio de todas las observaciones incluidas en su nodo terminal. En última instancia, cada árbol genera una partición del espacio

[2] Dichos árboles aplican reglas simples a los valores de las variables observables de forma secuencial para mejorar su capacidad predictiva. Por ejemplo, supongamos que queremos predecir si la calificación de los alumnos en un examen estandarizado es aprobatoria utilizando tan solo dos variables: las calificaciones promedio de su escuela el ciclo pasado y su estatura. El árbol de decisión tomaría en cuenta la variable de mayor poder predictivo en el primer nodo (supongamos que lo son las calificaciones promedio) y partiría el espacio paramétrico en dos (supongamos que es < 7 y $>= 7$). Posteriormente, en el siguiente nodo, generaría una nueva regla para la estatura de los niños.

de variables y asigna el valor promedio a todas las observaciones que se encuentran en la misma partición (Hastie *et al.*, 2013; Athey, 2018).

Los bosques aleatorios generan una gran muestra de árboles utilizando la técnica de *bootstraping* para generar submuestras. Posteriormente, calculan el valor promedio que cada observación recibió en cada árbol y esta es la predicción final (Hastie *et al.*, 2013). Tomar el promedio ayuda a evitar un importante problema en los árboles de decisión: el de ajustar excesivamente el modelo a los datos (*overfitting*).

Quizás el mayor inconveniente para el uso de este método en ciencias sociales es que es sumamente difícil interpretar sus resultados. A diferencia del método LASSO o los modelos AR, en los que interpretar los resultados de la regresión es relativamente sencillo, los bosques aleatorios fungen como una caja negra que poco pueden decirnos sobre cómo es que las variables independientes afectan a las dependientes (Varian, 2014).

Para efectos de este trabajo, consideraremos el bosque aleatorio para predecir el valor de la tasa de desocupación nacional de la ENOE (u_t) utilizando rezagos trimestrales de esta variable (u_{t-1}, u_{t-3}, u_{t-6}, u_{t-9}, u_{t-12}) y rezagos trimestrales del IGT (IGT_t, IGT_{t-1}, IGT_{t-3}, IGT_{t-6}, IGT_{t-9}, IGT_{t-12}). Nótese que, a diferencia de LASSO, no generamos interacciones no lineales de las variables para meterlas al modelo. La búsqueda de estas interacciones la hará el algoritmo.

6. Aprendizaje automático: resultados

6.1. *Errores a través del tiempo*

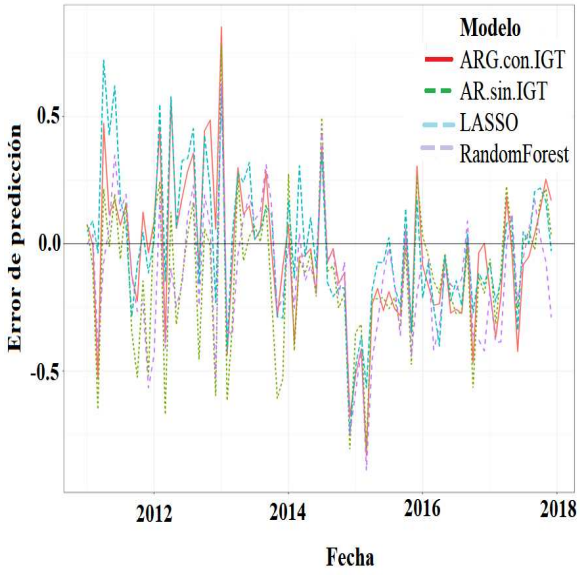
Antes de analizar qué tan precisos son los modelos en todo el periodo, es conveniente comprender cómo se comportan a través del tiempo. En la gráfica 4 observamos los errores de cada estimación para cada periodo.

En primera instancia, en la gráfica podemos notar que, en términos generales, todos los modelos cometen sistemáticamente errores similares. Es decir que, para prácticamente todos los meses, la diferencia entre los errores de los modelos corresponde a la magnitud del error y no su sentido (todos subestiman o sobrestiman la tasa de desempleo).

Quizás el modelo más interesante para analizar de esta gráfica sea el AR sin el IGT. Si bien es cierto que la dirección del error en

general no difiere de los otros modelos, podemos observar que dicho modelo suele subestimar con una magnitud mucho mayor la tasa de desempleo que el resto. Esta tendencia es mucho más clara para los meses previos a 2014.

Gráfica 4
Error de predicción por modelo



Fuente: Elaboración propia con datos de la ENOE y del IGT para la búsqueda "empleo + 'bolsa de trabajo' ".

Finalmente, llama la atención el periodo que va de mediados de 2014 a mediados de 2015, en el que todos los modelos subestiman consistentemente la tasa de desempleo. Si bien es cierto que, dado que utilizan prácticamente los mismos datos, no es del todo sorprendente que la pérdida en precisión durante un periodo resulte contagiosa, sí llama la atención la forma en que el método LASSO se recupera. En buena medida, se debe a que, para cada periodo, LASSO escoge cuáles serán sus estimadores, independientemente de lo que haya decidido en el periodo anterior.

En este sentido, LASSO puede ajustarse mucho mejor a momentos en los que la relación entre las variables se comporta de forma distinta.

6.2. *Error cuadrático medio*

Si bien es cierto que la gráfica 4 nos sirve para analizar las tendencias temporales de los estimadores, al presentar cuatro modelos en un periodo de 13 años, basándonos en ella resulta imposible poder identificar cuál es el modelo más eficiente. Por tal motivo, en esta sección analizaremos el error cuadrático medio durante todo el periodo para cada modelo.

En el cuadro 2 presentamos el error cuadrático medio de las predicciones para los cuatro modelos:

Cuadro 1
ECM por modelo

Modelo	*Error cuadrático medio*
AR con IGT	0.083
AR sin IGT	0.096
LASSO	0.077
Bosque aleatorio	0.084

Fuente: Elaboración propia con datos de la ENOE y del IGT para la búsqueda "empleo + 'bolsa de trabajo' ". Notas: Modelos estimados descritos en texto. AR se refiere a modelo autorregresivo

Como se puede observar, el modelo que comete errores de predicción, en promedio, más acertados es el método LASSO. Lo que podría indicarnos que la relación entre el índice de *Google Trends* y la tasa de desempleo no es necesariamente lineal, ya que entrenamos dicho modelo utilizando interacciones cúbicas.

En este sentido, los modelos de aprendizaje de máquina, en general, y el LASSO,en particular, sí parecen tener importantes ventajas predictivas sobre los métodos econométricos tradicionales. En el caso de LASSO, se lo podemos atribuír tanto a su flexibilidad para abordar interacciones no lineales entre los predictores como al uso de validación cruzada para ajustar el hiperparámetro λ y, de este modo, mejorar su capacidad predictiva.

También resulta interesante observar el comportamiento de los modelos autorregresivos. En primera instancia, el segundo modelo que mejor se comportó fue el AR cuando incluimos entre sus predictores al IGT. De hecho, la diferencia entre el ECM de este modelo y el ECM obtenido con el método LASSO no es tan grande. Es decir que,

pese a que la flexibilidad para representar relaciones no lineales sí trae consigo una mejora en la capacidad predictiva del modelo, un modelo lineal sencillo es capaz de "predecir el presente" razonablemente bien, sin la necesidad de utilizar hiperparámetros o validación cruzada.

Por otro lado, la diferencia del error cuadrático medio de los modelos AR cuando incluimos y cuando no incluimos el índice de *Google Trends* es considerable: el modelo autorregresivo sin el IGT es, por mucho, el modelo con la peor capacidad predictiva. Esto parece indicarnos que las nuevas fuentes de datos, en general, y que las búsquedas en *Google*, en particular, sí pueden ser de utilidad para desempañar labores de *nowcasting*, por lo menos en determinados contextos como el nuestro.

Finalmente, hay que comentar el desempeño del bosque aleatorio. Si bien es cierto que tiene un ECM mucho menor al modelo AR cuando no usamos el IGT, puede quizás resultar un tanto sorprendente que sea superado por el modelo autorregresivo con la información de *Google*. Aunque, en realidad, la precisión de ambos modelos no difiere sustancialmente, creemos que hay factores que podrían explicar dicho resultado.

Hay que recordar que el bosque aleatorio es un modelo más complejo y con más hiperparámetros a optimizar que el método LASSO. En nuestro caso, para efectos de este trabajo, sólo optimizamos dos hiperparámetros del bosque aleatorio: el tamaño de la muestra de predictores que se utilizan para construir cada árbol y el número de árboles. Si bien es cierto que las mejoras en precisión conforme encontramos la combinación de parámetros óptima para este problema son marginales, podríamos esperar una disminución en el ECM si ajustáramos otros hiperparámetros. En todo caso, esperaríamos que su precisión se mantuviera relativamente cercana a la que presenta el modelo AR con el IGT.

7. Conclusiones

El presente artículo tenía como propósito presentar algunos modelos propios del aprendizaje de máquina junto con nuevas fuentes de datos provenientes de *internet* y su utilidad para la ciencia económica en México. Concretamente, utilizamos tanto el IGT como el método LASSO y el de bosque aleatorio para intentar predecir el desempleo reportado por la ENOE.

En lo referente a los métodos, observamos que el LASSO sí presenta ganancias predictivas a un método propio de la econometría: los

modelos autorregresivos (AR). Además, el bosque aleatorio tuvo un desempeño apenas inferior al modelo autorregresivo. En este sentido, no se trata de que dichos métodos sean inherentemente superiores a aquellos que ya conocemos y que, por lo tanto, debamos preferir siempre a los primeros sobre los segundos. Por el contrario, concluimos que, debido a que se trata de algoritmos diseñados inherentemente para mejorar predicciones, los economistas debemos de conocerlos e incorporarlos al conjunto de herramientas que tradicionalmente utilizamos.

Por otro lado, es importante resaltar que la precisión de estos modelos y su capacidad predictiva son muy distintas cuando se utilizan en otros entornos. En este aspecto, creemos que conforme sean utilizadas para contestar nuevas preguntas iremos comprendiendo mejor cómo utilizarlas en economía y conoceremos sus alcances y limitaciones. Por su parte, el índice de *Google Trends* sí trajo consigo información de gran utilidad para mejorar las predicciones en la tasa de desempleo. Esto puede observarse al comparar la precisión del modelo autorregresivo cuando toma en cuenta el IGT y el del modelo autorregresivo cuando no lo hace.

La multiplicidad de nuevos datos que el *internet* trae consigo es de gran relevancia para la ciencia económica en México. Hoy tenemos acceso a una gran diversidad de fuentes de datos que pueden ser de utilidad para comprender mejor, desde la economía en particular y desde las ciencias sociales en general, nuestro entorno. Además, estas nuevas fuentes de datos no son solamente un insumo de gran utilidad para los científicos sociales. Por el contrario, los hacedores de políticas públicas pueden diseñar programas mucho más eficientes si cuentan con herramientas como estas.

La explosión de trabajos empíricos en economía que se ha dado desde hace ya algunos años atraviesa hoy por una nueva etapa en la que cada vez hay más datos (que, además, son sumamente granulares) y estamos conociendo nuevos métodos para analizarlos. Este artículo es tan solo una muestra de lo poderosas que pueden ser dichas herramientas aplicadas a la ciencia económica.

Agradecimientos

Agradecemos al director y dos dictaminadores anónimos por sus detallados comentarios y sugerencias. Todos los errores y omisiones son responsabilidad única de los autores, rmcampos@colmex.mx, selopab@gmail.com.

Referencias

Aguiar, E., H. Lakkaraju, N. Bhanpuri, D. Miller, B. Yuhas y K.L. Addison. 2015. Who, When, and Why: A Machine Learning Approach to Prioritizing Students at Risk of not Graduating High School on Time, *Proceedings of the 5th Learning Analytics and Knowledge Conference*.

Athey, S. 2017. Beyond prediction: Using big data for policy problems, *Science*, 355(6324): 483-485.

Athey, S. 2018. The Impact of Machine Learning on Economics (2018), en A. Agrawal, J. Gans y A. Goldfarb (comps.), *The Economics of Artificial Intelligence: An Agenda*, NBER, cap. 21, pp. 507-547, http://www.nber.org/chapters/c14009.

Baker, S. y A. Fradkin. 2017. The Impact of Unemployment Insurance on Job Search: Evidence from Google Search Data, *The Review of Economics and Statistics*, 99(5): 756-768.

Banco Mundial. 2014. Central America: Big Data in Action for Development, https://openknowledge.worldbank.org/handle/10986/21325.

Barjamovic, G., T. Chaney, K. Cosar y A. Hortaçsu. 2017. Trade, Merchants, and the Lost Cities of the Bronze Age, WP núm. 23992, NBER, Cambridge, http://www.nber.org/papers/w23992.

Beblavy, M., L. Kurekova y A. Thum. 2014. Using Internet Data to Analyse the Labour Market: A Methodological Enquiry, IZA DP núm. 8555, https://www.iza.org/publications/dp/8555/using-internet-data-to-analyse-the-labour-market-a-methodological-enquiry.

Belloni, A., V. Chernozhukov y C. Hansen. 2014. High-Dimensional Methods and Inference on Structural and Treatment Effects, *Journal of Economic Perspectives*, 28(2): 29-50.

Blumenstock, J. y D. Donaldson. 2013. How Do Labor Markets Equilibrate? Using Mobile Phone Records to Estimate the Effect of Local Labor Demand Shocks on Internal Migration and Local Wages, Proposal Summary for Application C2-RA4-205 (mimeo).

Bok, B., D. Caratelli, D. Giannone, A. Sbordone y A. Tambalotti. 2018. Macroeconomic Nowcasting and Forecasting with Big Data, *Annual Review of Economics*, 10(1): 615-643.

Carton, S., A. Mahmud, C. Cody, J. Helsby, Y. Park y R. Ghani. 2016. Identifying Police Officers at Risk of Adverse Events, Conference on Knowledge Discovery and Data Mining, http://www.kdd.org/kdd2016/papers/files/adf0832-cartonAemb.pdf.

Cavallo, A. y R. Rigobon. 2016. The Billion Prices Project: Using Online Prices for Measurement and Research, *Journal of Economic Perspectives*, 30(2): 151-178.

Chernozhukov, V., M. Demirer, E. Duflo e I. Fernandez-Val. 2018. Generic Machine Learning Inference on Heterogenous Treatment Effects in Randomized Experiments, WP, núm. 24678, NBER, Cambridge, http://www.nber.org/papers/w 24678.

Choi, H. 2010. Predicting Initial Claims for Unemployment Benefits, SSRN, https://ssrn.com/abstract=1659307.

Choi, H. y H. Varian. 2012. Predicting the Present with Google Trends, *Economic Record*, 88(s1): 29.

Einav, L., C. Farronato, J. Levin y N. Sundaresan. 2018. Auctions versus Posted Prices in Online Markets, *Journal of Political Economy*, 126(1): 178-215.

Einav, L., D. Knoepfle, J. Levin y N. Sundaresan. 2014. Sales Tax and Internet Commerce, *American Economic Review*, 104(1): 1-26.

Einav, L. y J. Levin. 2014a. The Data Revolution and Economic Analysis, *Innovation Policy and the Economy*, 14(1): 1-24.

Einav, L. y J. Levin. 2014b. Economics in the Age of Big Data, *Science*, 345(6210): 1-6.

Erel, I., L. Henny, C. Tan y M. Weisbach. 2018. Selecting Directors Using Machine Learning, WP núm. 24435, NBER, Cambridge, http://www.nber.org/papers/w24435.

Gerunov, A. 2014. Big Data Approaches to Modeling the Labor Market, publicado en *Proceedings of the International Conference on Big Data, Knowledge and Control Systems Engineering*, pp. 47-56.

Gleaser, E., H. Kim y M. Luca. 2017. Nowcasting the Local Economy: Using Yelp Data to Measure Economic Activity, WP núm. 24010, NBER, Massachusetts, http://www.nber.org/papers/w24010.

Global Pulse Lab. 2014. Mining Indonesian Tweets to Understand Food Price Crises, UN GLOBAL PULSE METHODS PAPER, february, https://www.unglobalpulse.org/projects/social-media-social-protection-indonesia.

Goel, S., J. Hofman, S. Lahaie, D. Pennock y D. Watts. 2010. Predicting consumer behavior with Web search, *Proceedings of the National Academy of Sciences*, 107(41): 17486-17490.

Hamermesh, D. 2013. Six Decades of Top Economics Publishing: Who and How? *Journal Of Economic Literature*, 51(1): 162-172.

Hastie *et al.* 2013. *An Introduction to Statistical Learning: with Applications in R*, Springer.

Hayashi, F. 2011. *Econometrics*, Princeton University Press.

Henderson, J., A. Storeygard y D. Weil. 2012. Measuring Economic Growth from Outer Space, *American Economic Review*, 102(2): 994-1028.

Hilbert, M. 2016. Big Data for Development: A Review of Promises and Challenges, *Development Policy Review*, 34(1): 135-174.

Horton, J. y P. Tambe. 2015. Labor Economists Get Their Microscope: Big Data and Labor Market Analysis, *Big Data*, 3(3): 130-137.

Hota, H.S., R. Handa y A.K. Shrivas. 2017. Time Series Data Prediction Using Sliding Window Based RBF Neural Network, *International Journal of Computational Intelligence Research*, 13(5): 1145-1156.

INEGI. 2017. Estadísticas a propósito del día mundial de Internet, Aguascalientes, México. http://www.inegi.org.mx/saladeprensa/aproposito/2017/internet2017_Nal.pdf

Inoue, Atsushi, Lu Jin y Barbara Rossi. 2017. Rolling window selection for out-of-sample forecasting with time-varying parameters, *Journal of Econometrics*, 196(1): 55-67.

Kennedy, R., D. Lazer y S. Wojcik. 2017. Improving election prediction internationally, *Science*, 355(6324): 515-520.

Lakkaraju, H., E. Guiar, C. Shan, D. Miller, N. Bhanpuri, R. Ghani y K.L. Addison 2015. A Machine Learning Framework to Identify Students at Risk of Adverse Academic Outcomes, *International Conference on Knowledge Discovery and Data Mining*.

Llorente, A., M. Garcia-Herranz, M. Cebrian y E. Moro. 2015. Social Media Fingerprints of Unemployment, *PLoS ONE* 10(5): e0128692.

Marinescu, I. 2017. The General Equilibrium Impacts of Unemployment Insurance: Evidence from a Large Online Job Board, *Journal of Public Economics*, 150(c): 14-29.

Marinescu, I. y R. Wolthoff. 2016. Opening the Black Box of the Matching Function: the Power of Words, *WP* núm. 22508, *NBER*, Cambridge, http://www.nber.org/papers/w22508t.

Mullainathan, S. y Z. Obermeyer. 2017. Does Machine Learning Automate Moral Hazard and Error? *American Economic Review: Papers and Proceedings*, 107(5): 476-480.

Mullainathan, S. y J. Spiess. 2017. Machine Learning: An Applied Econometric Approach, *Journal of Economic Perspectives*, 31(2): 87-106.

Rogers, S. 2016. What is Google Trends data??and what does it mean? https://medium.com/google-news-lab/what-is-google-trends-data-and-what-does-it-mean-b48f07342ee8.

Rundle, A., M. Bader, C.A Richards, K.M. Neckerman y J.O. Teitler. 2011. Using Google street view to audit neighborhood environments, *American Journal of Preventive Medicine*, 40(1): 94-100.

Sara, N.B., R. Halland, C. Igel y S. Alstrup. 2015. High-School Dropout Prediction Using Machine Learning: A Danish Large-scale Study, en M. Verleysen (comp.), *Proceedings, ESANN 2015: 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, i6doc.com, pp. 319-324.

Stephens-Davidowitz, S. 2017. *Everybody Lies: Big Data, New Data, and What the Internet Can Tell Us About Who We Really Are*, HarperCollins Publishers, Nueva York.

Taddy, M. 2018. The Technological Elements of Artificial Intelligence, WP núm. 24301, NBER, Cambridge, http://www.nber.org/papers/w24301.

Tuhkuri, J. 2015. Big Data: Do Google Searches Predict Unemployment?, tesis de maestría, Universidad de Helsinki, https://helda.helsinki.fi/handle/10138/155258.

Varian, H. 2014. Big Data: New Tricks for Econometrics, *Journal of Economic Perspectives*, 28(2): 3-28.

Material complementario versión digital

ANEXO A: Rolling window, LASSO, y Random Forest en R

R es un lenguaje de programación especializado en manipulación de datos y análisis estadístico. En él, es posible trabajar con todo tipo de objetos ─bases de datos (conocidos como *data frames*), matrices, vectores y escalares, entre otros─ de forma simultánea. Se trata de un *software* libre, con una gran comunidad de usuarios a nivel global, en el que potencialmente cualquiera podría desarrollar paquetes (es decir, un conjunto de funciones) dedicadas a realizar tareas determinadas.

Por estos motivos es que R es quizás una de las plataformas más populares a nivel profesional para implementar algoritmos de aprendizaje de máquina. Además, debido a dicha popularidad, R cuenta ya con una gran cantidad de paquetes para hacer prácticamente cualquier tipo de análisis de datos. De hecho, es relativamente común que, cuando se propone una nueva estrategia de análisis, se presente también un paquete estadístico en R para implementarlo.

Pese a que hasta la fecha la comunidad de ciencias sociales no ha adoptado de forma homogénea el uso de este lenguaje, su número de usuarios entre economistas, politólogos y sociólogos, entre otros, sí va en aumento, tanto en México como en el resto del mundo. Por este motivo es que resulta conveniente comprender este lenguaje y agregarlo al conjunto de herramientas estadísticas que usamos los economistas.

En este anexo, explicamos el algoritmo para entrenar, generar y evaluar los modelos de LASSO y bosque aleatorio en R utilizando la ventana cambiante (*rolling window).* Además de explicar más a detalle los algoritmos, presentamos también el código en R que utilizamos

y lo explicamos paso por paso. Antes de comenzar, vale la pena recalcar que para entrenar tanto al LASSO como al bosque aleatorio utilizamos la misma estructura de código para implementar la ventana cambiante para su evaluación, por lo que esta podría ser un poco repetitiva. Comenzaremos explicando cómo funciona el algoritmo de la ventana cambiante utilizando un modelo AR y, posteriormente, nos centraremos en los modelos más propios del aprendizaje de máquina.

A1. Elementos básicos de R

Es importante comenzar algunos elementos básicos de R. En primera instancia, es importante aclarar que el operador "<-" se utiliza para asignar valores, de tal suerte que, para declarar que "a=3", escribiría "a<-3". Pese a que desde hace un par de años R ha comenzado a reconocer el operador "=", sigue siendo recomendable usar la flecha[1]. Otro elemento importante es el operador "c()", que significa concatenar. Este operador nos permite crear vectores, de tal suerte que si queremos asignarle a la variable "a" el vector [3,4,7,9], escribiríamos "a<-c(3,4,7,9)".

Finalmente, es importante hablar de los índices. R es un lenguaje base uno, lo que significa que comienza a indexar los elementos de matrices y vectores a partir del número 1 (esta distinción es importante puesto que hay otros lenguajes, como Python, que lo hacen a partir del cero). Ahora bien, para acceder a los elementos de un objeto basta con poner, inmediatamente al lado del nombre del objeto, entre corchetes, el número de elemento al que se desee acceder. Por ejemplo, si queremos obtener el segundo elemento del vector "a" que

[1] Hay que enfatizar que la dirección de la flecha importa. Siguiendo con el ejemplo anterior, es equivalente escribir "a<-3" que "3->a", pero escribir "3<-a" o a->3" sería incorrecto, pues no se le puede asignar un valor a un número.

definimos anteriormente (recordemos que a=[3,4,7,9]) basta con escribir “a[2]”, lo cual nos regresaría el número 4. Si se trata de una matriz, los índices van separados por una coma, el primer elemento se refiere al renglón y el segundo a la columna (ej. B[2,4] nos regresaría el valor que la matriz B tenga en su segundo renglón y en su cuarta columna).

A.2 Ventana cambiante

En esta sección presentaremos el algoritmo de ventana cambiante. Como se mencionó anteriormente, este se utiliza tanto para la búsqueda de los hiperparámetros apropiados (a lo que llamamos “entrenamiento del modelo”) como para la evaluación de los modelos, por lo que entender su estructura es esencial. Utilizaremos como ejemplo los modelos AR, tanto el que incluye al IGT como al que lo omite. Esta sección del código no la utilizamos durante nuestros cálculos, la presentamos aquí para mostrar la estructura de la ventana móvil.

En primera instancia, es importante describir nuestros elementos de trabajo. Contamos con una base de datos, a la que hemos decidido llamar “nacional” (puesto que información a nivel nacional) en la que se encuentran todos nuestros datos. En ella tenemos a la tasa de desocupación (a la que llamamos “desocupación_nacional”) con sus respectivos rezagos trimestrales (que reciben el nombre “desocupación_nacional_lx”, en donde “x” representa el número de meses por el que fueron rezagadas). Por otro lado, tenemos el IGT al que hemos llamado “indice” con sus respectivos rezagos, nombrados bajo la misma lógica que los rezagos de la tasa de desocupación. Hay que recalcar que, para que este código funcione, las observaciones deben de estar ordenadas por fecha de forma ascendente.

El primer paso para implementar un algoritmo de ventana móvil consiste en declarar el ancho de la ventana y el número de ventanas:

w_size<-60

n_windows<-nrow(nacional) – 60

En nuestro caso, utilizamos una ventana de 60 meses. Notemos que para definir el número total de ventanas utilizamos la función “nrow()”, que nos devuelve el número de renglones en una matriz o una base de datos.

Ya que tenemos claro el ancho de las ventanas y el número de ventanas que podemos generar, lo que sigue es generar un ciclo (*loop*) que vaya recorriendo los renglones de la base, corriendo el modelo AR y después guardando las predicciones de cada iteración. A continuación, presentamos el código íntegro para hacerlo:

```
forecasts = foreach(i=1:n_windows, .combine = rbind) %do%{
nacional_in = nacional[i:(w_size + i - 1), ]
nacional_out = nacional[w_size + i, ]
m1 = lm(desocupacion_nacional ~ desocupacion_nacional_l1 +
desocupacion_nacional_l12 + indice, data = nacional_in)
f1 = predict(m1, nacional_out)
m2 = lm(desocupacion_nacional ~ desocupacion_nacional_l1 +
desocupacion_nacional_l12, data = nacional_in)
f2 = predict(m2, nacional_out)
return(c(f1, f2))}
```

En primera instancia, es conveniente tener una idea general de lo que estamos haciendo: al objeto “forecasts” le estamos asignando los valores que regrese el ciclo que declaramos inmediatamente después. Este ciclo, como podrá observarse, lo declaramos con la función “foreach”, a la que hay que otorgarle un vector de valores (“i=1:n_windows”). Notemos que el operador “:” sirve para generar un vector que vaya desde lo que se encuentre de su lado

izquierdo ─en este caso “i”─ hasta el valor que se encuentre del lado derecho ─en este caso “n_windows”─. Además, debemos de especificar a la función la estructura de datos que queremos que nos regrese (“.combine = rbind”). Este último elemento le indica a la función que lo que esperamos recibir es una matriz cuyas filas se van a ir llenando en cada iteración. Finalmente, el operador “%do%” le indica a la función que haga sus cálculos de forma secuencial (“%dopar%” le indicaría que los hiciera en paralelo).

Lo primero que estamos haciendo es generar las submuestras. La variable *nacional_in* está tomando todas las observaciones que se encuentran en la base de datos *nacional* desde la observación *i*(es decir, el número de iteración) hasta la observación “n_windows + i -1”. Por su parte, la variable *nacional_out* toma el valor de la observación inmediatamente posterior. Esto nos permitirá posteriormente evaluar las predicciones del modelo.

El resto del código consiste en correr la regresión y después hacer las predicciones. Recordemos que un modelo AR no es más que un problema de MCO con valores rezagados de la serie de tiempo. Podemos correr regresiones con la función “lm()” (por “modelo lineal” en inglés), a la que le tenemos que decir la fórmula de la regresión que queremos correr (si queremos hacer una regresión de *x* y *z* sobre *y* escribiríamos “y~x+z”) y el nombre del objeto del que obtendremos los datos (en nuestro caso, lo hacemos con la base de datos “nacional_in” que generamos en cada iteración).

La función “lm()” genera varios valores, entre ellos el valor de los coeficientes de la regresión. Dichos valores los guardamos en el objeto “m1”. Posteriormente, utilizamos la función “predict()” para, con los valores que guardamos en “m1”, hacer una predicción del valor que tomará la tasa de desempleo dados los valores de sus predictores. Notemos que basta con que “nacional_out” contenga las mismas variables que utilizamos en la regresión

para que la función “predict()” genere la predicción. El valor de la predicción se lo asignamos a “f1”.

Como se mencionó anteriormente, se utilizaron ambos modelos AR por lo que lo que sigue en el código es exactamente igual: corremos la regresión (ahora sin el IGT), le asignamos sus valores a “m2”, hacemos la predicción para “nacional_out” y asignamos este valor a “f2”. Posteriormente, utilizamos “return(f1,f2)” para indicarle a la función que lo que queremos que nos devuelva sean las predicciones.

Ahora, “forecasts” es una matriz que contiene todas las predicciones hechas mediante la ventana móvil, su primera columna contiene las predicciones para el modelo AR con el IGT y la segunda para el AR sin el IGT. Generar un vector con los errores de predicción para cada modelo y cada observación es sencillo:

```
e1 = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,1]
e2 = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,2]
```

Utilizamos la función “tail()” para quedarnos con las últimas *n* observaciones de “nacional” (en este caso *n* se refiere al número de renglones de “forecasts”) y la variable “desocupación nacional”. Finalmente, a este vector le restamos el vector generado por, respectivamente, la primer y la segunda columna de “forecasts”. De este modo, “e1” contiene ahora los errores de predicción del modelo AR con el IGT y “e2” el del AR sin el IGT. Como veremos más adelante, ahora que tenemos estos vectores de errores podemos generar gráficas, calcular el ECM de cada modelo y más.

A.3 Parametrizando LASSO

A continuación, presentamos el código utilizado para buscar el hiperparámetro λ óptimo para el LASSO. Sin embargo, antes de hacerlo, hay que generar primero la base de datos que utilizaremos pues, como se mencionó en la sección 5, generamos todas las interacciones cúbicas entre el IGT y sus rezagos con la tasa de desocupación y sus rezagos.

```
polinomializable <- nacional[ , -which(names(nacional) %in% c("Mes","fecha",
"Estado", "hombres_desocupacion_nacional" ,"hombres_participacion_nacional",
"mujeres_desocupacion_nacional" ,"mujeres_participacion_nacional",
"desocupación_nacional"))]

pol <- as.matrix(polinomializable)

predictores <- scale(poly(pol, degree = 3, raw = T))
```

En primera instancia, generamos la matriz "polinomializable" removiendo de la base de datos "nacional" las columnas que no nos interesan. La función "which" nos ayuda a seleccionarlas y el signo "-" indica que lo que se hará con esas columnas será quitarlas. Finalmente, la función "as.matrix()" nos permite convertir la base de datos "polinomializable" en una matriz. Recordemos que, aunque ambos tengan el mismo formato, para R son objetos distintos con propiedades diferentes. Finalmente, utilizamos la función "poly()" para generar la matriz "predictores" con todas las interacciones cúbicas para las columnas de la matriz "pol". A esta matriz, le aplicamos la función scale para estandarizar todas sus columnas.

```
response <- as.vector(nacional$desocupacion_nacional)

lambdas <- 10^seq(10, -2, length = 100)

mejor_lambda <- -1

error_min <- 50000
```

Generamos también otros objetos que serán necesarios en el proceso. Para correr el modelo LASSO utilizaremos el paquete *glmnet* y la función homónima. A diferencia de la función "lm()", "glmnet()" no acepta bases de datos (*dataframes*) como elementos de entrada, por lo

que debemos de trabajar con matrices. Por este motivo, generamos el vector “response” que contiene todos los datos de desocupación nacional incluidos en la base “nacional”.

Por otro lado, generaremos un vector de distintos valores para λ con el objetivo de iterar sobre de él y encontrar la que minimice el ECM. Aunque no hay una motivación teórica acerca de qué valores de λ debemos probar, este vector es el más utilizado. Finalmente, definimos un error mínimo artificial, su utilidad se hará evidente en cuanto analicemos el resto del código.

A continuación, presentamos el ciclo utilizado para buscar el hiperparámetro λ que minimice el ECM. En términos generales, estamos corriendo el mismo ciclo de ventana móvil explicado en la subsección anterior para cada posible valor de λ en el vector “lambdas”.

```
for (l in lambdas){forecasts = foreach(i=1:n_windows, .combine = rbind) %do%{
x_in = as.matrix(predictores[i:(w_size + i - 1), ])
y_in = response[i:(w_size + i - 1)]
x_out = predictores[w_size + i, ]
m1 = glmnet(x=x_in, y=y_in, lambda = l, alpha = 1)
f1 = predict(m1, t(x_out), s=l, type="response")
return(c(f1))}
error = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,1]
rmse = 1000 * sqrt(mean(error ^ 2))
if (rmse<error_min){
mejor_lambda <- l
error_min<-rmse }}
```

En la primera línea del código, estamos indicándole a R que corra el ciclo de ventana móvil que vimos anteriormente ─ahora aplicado a LASSO─ para cada valor *l* en el vector

"lambdas". El objeto "forecasts", la función "foreach" y el operador "%do%" se utilizan exactamente igual que antes.

Por otro lado, para utilizar la función "glmnet" necesitamos una matriz de predictores y un vector con los valores de la variable que queremos predecir. Por este motivo generamos los objetos "x_in" y "x_out" ─predictores dentro y fuera de la ventana, respectivamente─ así como "y_in", que contiene los valores de la variable objetivo. Notemos que el criterio de asignación de renglones se da igual que antes: desde *i* (el número de iteración) hasta *w_size +i-1*.

Nuevamente, en "m1" guardamos el resultado del modelo LASSO que obtenemos de correr la función "glmnet()". A esta función tenemos que darle varios argumentos: *x*, los predictores ("x_in" en nuestro caso); *y*, la variable objetivo ("y_in", en nuestro caso); *lambda*, el parámetro de penalización (al que nosotros le asignamos "l" que es la λ que estamos tomando en cada iteración) y *alpha*, que siempre es igual a uno para que el modelo sea un LASSO[2].

El paso restante consiste en tomar los vectores de predicciones que emergieron con el valor de λ de cada iteración para generar el vector de errores y, posteriormente, el ECM. Finalmente, si el ECM de esa iteración es menor a "error_min"[3] y, de ser así, asignamos el valor de "rmse" a "error_min" y el de "l" a "mejor_lambda". Como resultado, obtendremos el valor de λ que minimiza el ECM, calculado con el algoritmo de ventana móvil, para el método LASSO.

[2] La función "glmnet" nos sirve para correr modelos más generales que LASSO. Si definieramos α=0, nos quedaríamos con otro modelo para grandes dimensiones llamado Ridge.
[3] En la primera iteración es muy factible que lo sea pero, a partir del segundo valor *l*, estaremos comparando los errores entre los modelos.

A.4 Parametrizando el bosque aleatorio

En términos generales, el algoritmo para parametrizar el bosque aleatorio funciona del mismo modo que el de LASSO: generamos vectores de posibles valores de los hiperparámetros, corremos un ciclo en el que, para cada valor o combinación de valores, calculemos el valor de los errores utilizando el ciclo de ventana móvil y, finalmente, nos quedamos con los valores de los hiperparámetros que minimicen el ECM.

mejor_m <- -1

mejor_t <- -1

error_min <- 10000000000

mvector <- ncol(nacional)/c(1:19)

tvector <- seq(from=500, to=550, by=5)[4]

set.seed(1993)

Hay que mencionar que los bosques aleatorios son modelos muy populares puesto que simultáneamente son muy flexibles (es decir, podemos ajustar muchos hiperparámetros) y no requieren ser ajustados de forma exhaustiva. Para fines del presente trabajo, nosotros decidimos enfocarnos en dos hiperparámetros del bosque aleatorio: el número de variables que se seleccionan aleatoriamente para construir cada árbol individual (*m*, en el vector "mvector") y el número total de árboles (*t*, en el vector tvector). Finalmente, la función "set.seed()" nos permite fijar la distribución aleatoria de los datos para asegurarnos que el ejercicio es replicable.

for (t in tvector){for (m in mvector){

[4] *t*=500 es el valor que la función "RandomForest" introduce por default. Para poder hacer una búsqueda amplia sobre el valor óptimo corrimos los ciclos para distintos vectores "tvector". La idea es primero generar un vector con mayor amplitud aunque menos granular (ej. Nosotros comenzamos con *tvector = seq(400,1500, by=150)*) e ir concentrándonos en las regiones en las que el ECM se minimice.

```
forecasts = foreach(i=1:n_windows, .combine = rbind) %do%{
nacional_in = nacional[i:(w_size + i - 1), ]
nacional_out = nacional[w_size + i, ]
m4 = randomForest(desocupacion_nacional ~ desocupacion_nacional_l1  +
desocupacion_nacional_l3 + desocupacion_nacional_l6 + desocupacion_nacional_l9 +
desocupacion_nacional_l12 + indice + indice_l1 + indice_l3 + indice_l6 + indice_l9 +
indice_l12, data=nacional_in, mtry=m, ntree=t)
f1 =predict(m4, nacional_out)
return(c(f1)) }
error = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,1]
rmse = 1000 * sqrt(mean(error ^ 2))
if (rmse<error_min){
mejor_m <- m
mejor_t <- t
error_min<-rmse}}}
```

Como la estructura del código es muy similar a las que hemos visto anteriormente, no nos detendremos en explicar todos los pormenores y nos concentraremos únicamente en las principales novedades. La primera diferencia que salta a la vista es que ahora se trata de tres ciclos, no de dos como en el caso de LASSO. Esto responde al espacio de hiperparámetros: ahora tenemos dos (t y m) cuando antes solo teníamos uno (λ). Además, cabe mencionar que, al igual que la función "lm()", la función "randomForest()" sí acepta como entrada una base de datos, por lo que no es necesario transformar a la base "nacional" en una matriz, como sí lo fue para parametrizar LASSO.

Comentemos la función "randomFores()", pues se trata de una función tan flexible como el modelo. En primera instancia, hay que hacer explícito cuál es la variable que queremos predecir, cuáles los predictores y de qué base de datos los estamos sacando, con la misma sintaxis en que lo hicimos para el AR. En nuestro caso, es necesario declarar también los

valores de los hiperparámetros que decidimos afinar: “ntree” y “mtry”. Notemos que, al igual que en LASSO, asignamos a los hiperparámetros el valor que adquieren en cada iteración.

Finalmente, al igual que en la subsección anterior, obtenemos el vector de predicciones, lo usamos para calcular el vector de errores y calculamos el ECM para cada combinación de hiperparámetros. Haciendo la comparación de cada ECM, nos quedamos con los valores de *t* y *m* que lo minimizan. Guardamos estos valores en “mejor_m” y “mejor_t”.

A.5 Generando la base de todos los modelos

Una vez que encontramos los valores óptimos para λ, *m* y *t* (guardados en “mejor_lambda”, “mejor_m” y “mejor_t”, respectivamente), queremos construir una base de datos con los errores de los cuatro modelos para poder compararlos. Para hacerlo, utilizamos el ciclo que que vimos en la sección A2 para explicar la ventana móvil, pero añadiéndole los dos modelos faltantes:

```
set.seed(1993)
forecasts = foreach(i=1:n_windows, .combine = rbind) %do%{
nacional_in = nacional[i:(w_size + i - 1), ]
nacional_out = nacional[w_size + i, ]
x_in = as.matrix(predictores[i:(w_size + i - 1), ])
y_in = response[i:(w_size + i - 1)]
x_out = predictores[w_size + i, ]
# AR con IGT #
m1 = lm(desocupacion_nacional ~ desocupacion_nacional_l1 +
desocupacion_nacional_l12 + indice, data = nacional_in)
f1 = predict(m1, nacional_out)
#AR sin IGT#
m2 = lm(desocupacion_nacional ~ desocupacion_nacional_l1 +
desocupacion_nacional_l12, data = nacional_in)
```

```
f2 = predict(m2, nacional_out)
# LASSO #
m3 = glmnet(x=x_in, y=y_in, lambda = mejor_lambda, alpha = 1)
f3 = predict(m3, t(x_out), s=l, type="response")
# Random Forest #
m4 = randomForest(desocupacion_nacional ~ desocupacion_nacional_l1  +
desocupacion_nacional_l3 + desocupacion_nacional_l6 + desocupacion_nacional_l9 +
desocupacion_nacional_l12 + indice + indice_l1 + indice_l3 + indice_l6 + indice_l9 +
indice_l12, data=nacional_in, mtry=mejor_m, ntree = mejor_t)
f4 =predict(m4, nacional_out)
return(c(f1, f2, f3, f4))
setTxtProgressBar(pb, i)}
```

Notemos que estamos brindando los valores óptimos de los hiperparámetros que encontramos anteriormente para, en este ciclo, quedarnos con las mejores predicciones que posibles. Ahora, el objeto “forecasts” contendrá la matriz con las predicciones de los cuatro modelos. Con él, podemos construir los vectores de errores para, por ejemplo, graficar su comportamiento en el tiempo (ver gráfica 4):

```
e1 = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,1]
e2 = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,2]
e3 = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,3]
e4 = tail(nacional[ ,"desocupacion_nacional"], nrow(forecasts)) - forecasts[ ,4]
df = data.frame("date"=tail(nacional$fecha, n_windows), "AR con IGT" = e1, "AR sin
IGT" = e2, "LASSO" = e3, "RandomForest" = e4)
mdf = melt(df,id.vars = "date")
names(mdf)[2] <- "Modelo"
```

Primero, al igual que como lo hicimos en la sección A.2, construimos los vectores de errores de predicción de cada modelo. Con ellos, construimos una base de datos y tomamos sus

fechas de la base “nacional”. Hacemos esto con la función “dataframe()”, a la que le indicamos los datos y los nombres de las columnas. Finalmente, generamos la base “mdf” con la función “melt()”, que nos apila las columnas por fecha para generar las series de tiempo. Este último paso es fundamental para generar la gráfica. Es importante recalcar que la base “mdf” contiene tres columnas: fecha, error y tipo de modelo.

Por otro lado, calcular el ECM de cada modelo es relativamente sencillo utilizando los vectores de errores:

```
rmse_ar_gt =sqrt(mean(e1 ^ 2))

rmse_ar =  sqrt(mean(e2 ^ 2))

rmse_lasso = sqrt(mean(e3 ^ 2))

rmse_rf =  sqrt(mean(e4 ^ 2))
```

A6. Gráficas

En términos generales, pueden generarse gráficas en R utilizando ya sea el motor base (con la función “plot()”) o a través del paquete *ggplot2*. Debido a su enorme flexibilidad, esta última opción es, por mucho, la más popular para los usuarios de R. Por este motivo, esta es la opción que utilizamos para generar la gráfica.

```
gr <- ggplot(data = mdf) + geom_line(aes(x = date, y = value, linetype = Modelo, color = Modelo)) +
      labs(title="", x="Fecha", y="Error de predicción", variable="Modelo") +
      theme_bw() + geom_hline(yintercept = 0)+
      theme(title = element_text(size=20, face="bold"),
            axis.text=element_text(size=20),
            axis.title.y =element_text(size=20),
            legend.position = c(0.89, .865),
```

legend.text=element_text(size=15))

La sintaxis para generar gráficas en *ggplot2* es relativamente sencilla: primero utilizas la función "ggplot()" para declarar elementos principales de la gráfica y posteriormente, mediante el símbolo "+", se van agregando nuevos elementos. En nuestro caso, a la función "ggplot()" solo le indicamos que los datos necesarios para generar la gráfica se encuentran en la base de datos llamada "mdf".

Posteriormente, utilizamos la función "geom_line()" para especificar que lo que queremos graficar son líneas (podríamos usar, por ejemplo, "geom_point()" o "geom_bar()" para graficar puntos o barras, respectivamente). Dentro de "geom_line()" declaramos los elementos de la gráfica usando "aes()": indicamos qué queremos en cada eje y cuáles son los diferentes grupos dentro de la base de datos (en nuestro caso, los grupos corresponden a los modelos predictivos).

Posteriormente, con la función "labs()" declaramos el texto que queremos que aparezca tanto en el título como en los ejes, así como el título de la leyenda para diferenciar a los grupos. Utilizamos "theme_bw()" para cambiar el color del fondo de la gráfica (que, por *default,* es gris) y, con "geom_hline()" generamos la línea roja en el eje *x*. Finalmente, la función "theme()" nos permite modificar múltiples elementos de la gráfica: en nuestro caso, el tamaño de texto del título y del nombre de los ejes, así como la posición de la leyenda.

Apéndice B

Sin IGT

Modelo	ECM
$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-12} + \epsilon_t$	0.096

(el que utilizamos)

$$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-6} + \beta_3 y_{t-12} + \epsilon_t \qquad 0.096$$

$$y_t = \beta_1 y_{(t-1)} + \beta_2 y_{(t-3)} + \beta_3 y_{(t-6)} + \beta_4 y_{(t-9)} + \beta_5 y_{(t-12)} + \epsilon_t \qquad 0.096$$

Con IGT

Modelo	ECM
$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-12} + IGT_t + \epsilon_t$	0.084

(el que utilizamos)

$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-6} + \beta_3 y_{t-12} + IGT_t + \epsilon_t$	0.084
$y_t = \beta_1 y_{(t-1)} + \beta_2 y_{(t-3)} + \beta_3 y_{(t-6)} + \beta_4 y_{(t-9)} + \beta_5 y_{(t-12)} + IGT_t + \epsilon_t$	0.081