FILMSTRADAMUS 3000

Recomendador de programación para salas de cine

Ana Zumalacárregui Pérez
Dhani G. Montoya Rojas
José Tellechea Mora
Mario A. Bolaños Amaya
Pablo Fernández Rodríguez

EOI Escuela de organización industrial

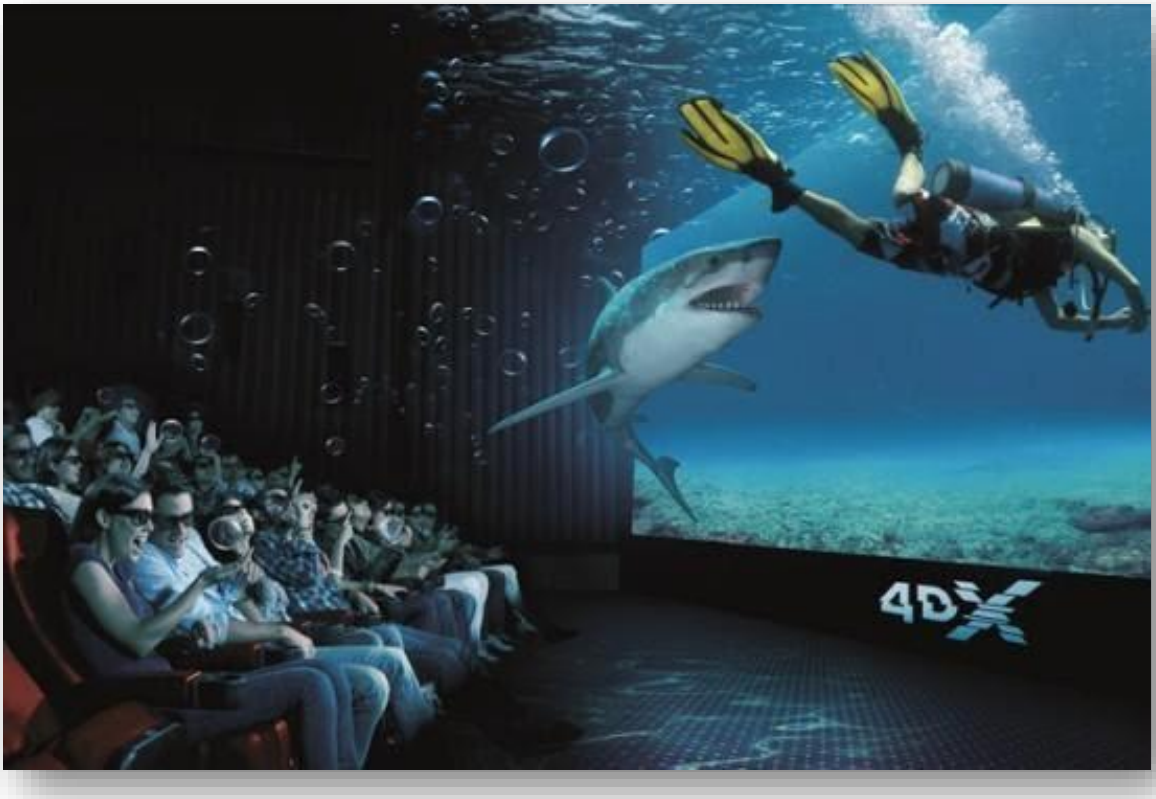
Agradecemos a nuestro tutor Daniel Burrueco su guía y apoyo constante durante la elaboración de este proyecto.

Índice

La industria, hoy.

La industria cinematográfica ha sido hasta hace aproximadamente diez años la primera en importancia del sector del entretenimiento. En el año 2019, facturó globalmente 29.380 millones de euros.

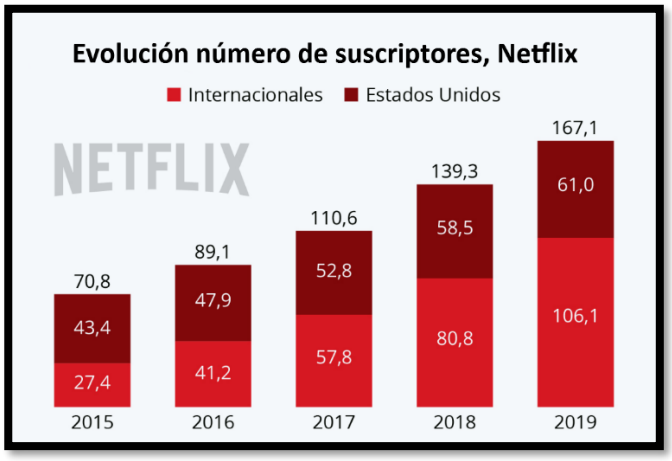
Para que una película rentabilice la inversión que en ella se realiza, se deben dar dos circunstancias: que el producto sea del gusto del público, y de no menos importancia, que llegue a los espectadores a través del canal adecuado. Por tanto, existen dos actores principales por el lado de la oferta cinematográfica: el estudio o productora que crea el producto original, y el exhibidor, que lo lleva hasta el espectador. El elemento central es la producción de la película, proceso en el cual no interviene el exhibidor. Pero por muy buenas películas que se produzcan, si no cuentan con los canales de distribución adecuados, el público no tendrá acceso a ellas, convirtiéndolas en irrelevantes, o incluso en fracasos comerciales, pudiendo provocar grandes pérdidas a los productores.



Tradicionalmente, la exhibición cinematográfica se ha dado en las salas de cine de forma casi exclusiva. El advenimiento del video doméstico y de otros formatos digitales en las décadas de los 80 y 90 del siglo pasado amenazó en primera instancia a la sala tradicional, pero finalmente se llegó a una convivencia en la cual el estreno de las producciones y su ciclo principal de exhibición permaneció en el cine, estableciéndose un canal alternativo, si bien no coincidente en el tiempo, a través del vídeo, del dvd, e incluso de la televisión, que posibilitó de hecho que las películas pudieran disfrutar de un segundo ciclo comercial productivo, que no se daba con anterioridad a la llegada del formato doméstico.

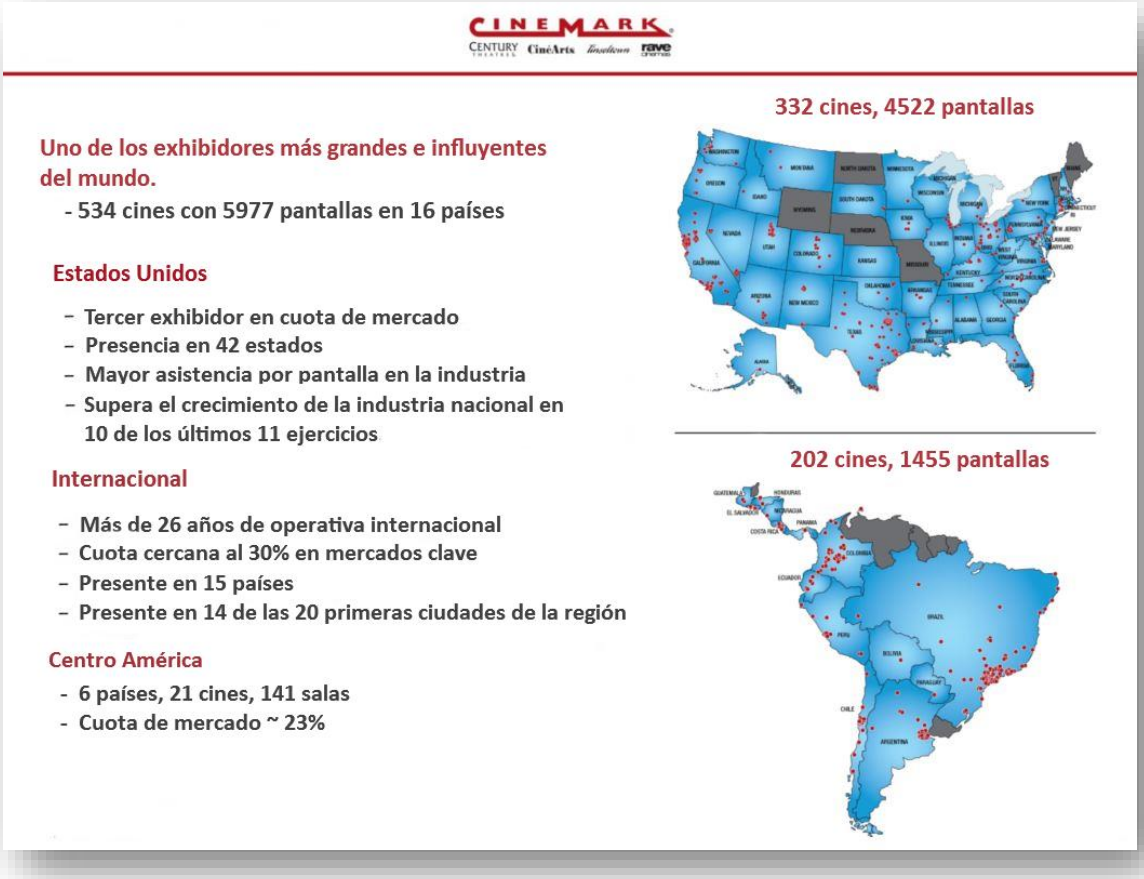
En los últimos quince años, sin embargo, se ha acelerado un proceso mucho más amenazador para el exhibidor tradicional. Factores tanto técnicos como socioeconómicos han posibilitado este fenómeno. Por un lado, la generalización de la banda ancha, la mejora de las capacidades de los dispositivos móviles y receptores de televisión, que se han abaratado a la vez que han multiplicado sus prestaciones. Por otro, la irrupción de las plataformas de streaming y OTT , que actúan tanto como productores como exhibidores no sólo ya en contenidos tradicionalmente entendidos como televisivos (documentales, series, etc.) sino también en producciones cinematográficas, entendiendo el término en su acepción "clásica". El caso de la producción "Roma", ganadora en el Festival de Venecia y nominada a diez premios Oscar, pero distribuida exclusivamente por Netflix, y con un ciclo de exhibición en sala meramente testimonial, es un claro ejemplo de la transformación del sector y de los cambios ya acontecidos , impensables hace tan solo unos años.

Se puede afirmar, por tanto, que la industria exhibidora tradicional se encontraba, a finales de 2019, en un momento de cambios estructurales, y ante un rival, las plataformas digitales (Netflix, HBO, Amazon, Hulu, la lista es larga), que cuenta con una ventaja tecnológica e incluso financiera indudable.
En 2020, con la eclosión del Covid-19 a nivel mundial, el escenario se ha vuelto crítico. El cierre obligado de las salas durante gran parte del año, y la ocupación parcial (entre el 20 y el 40% del aforo disponible) decretada bajo los protocolos de seguridad en reapertura, han hundido las facturaciones, convirtiéndose este año en el peor que se recuerda en el sector.



Caso de uso.

El caso que nos ocupa en este proyecto nace de nuestra relación con Cinemark a través de uno de los miembro del grupo, empleado de la compañía. Cinemark, fundada en 1997, con sede en Plano, Texas, es el tercer exhibidor mundial, una compañía listada en la bolsa de Nueva York, con una capitalización bursátil de aproximadamente 1.020 millones de dólares. En nuestro caso colaboramos con su sucursal de Centro América, región en la cual está presente en 6 países, con un total de 21 cines con 141 salas. En total en la región existen 132 cines (con 745 salas), lo cual implica que Cinemark cuenta con el 16% de los cines y el 19% de las salas.



Definición del problema.

Antes de la caída sin precedentes de los ingresos en todo el sector exhibidor, Cinemark había visto en el último año (2019) un ligero descenso de su cuota de mercado.

Los motivos de esta pérdida de negocio percibidos por la empresa caían tanto fuera como dentro del ámbito de control de la misma. Entre aquellos, el endurecimiento de la competencia con la apertura de nuevos cines en la región. Entre los que sí pertenecen al área de control de la gestión propia, se detectaron diferentes puntos de potencial mejora. Obviamente, nuestra aportación debía centrarse en el segundo grupo.

Asimismo, las soluciones o herramientas que ofreciéramos debían reunir otra serie de requisitos, en vista de la complicada coyuntura de negocio. Una solución de coste contenido pero alto potencial de beneficio materializable en un período relativamente corto de tiempo, que no requiriese de una gran inversión inicial, y que una vez establecida y funcionando en la región, pudiera (si así lo considerase oportuno la casa matriz) implantarse en otras regiones.

Tras un análisis de la actividad de Cinemark y un estudio de su proceso de negocio desde diferentes puntos de vista, detectamos un apartado del mismo que reunía las condiciones arriba mencionadas: la programación de las películas en sala.

Cuando una película llega a la pantalla, el exhibidor la emplaza en unas salas y horarios determinados; igualmente, decidirá la duración del ciclo de exhibición de la película en sala en función de su desempeño.

Algo que la empresa comprueba semana tras semana, y a lo que dedica un considerable esfuerzo, es que haciendo ajustes en su programación, reprogramando, se observa un aumento de la cuota de mercado, entre la asistencia a sesiones programadas "en frío" y a las posteriores a los reajustes de reprogramación (basados, evidentemente, en la información observada en aquellas), de entre uno y tres puntos porcentuales.

Como se ha señalado antes, la programación es una actividad totalmente controlable por Cinemark. Si se estimase con mayor precisión lo que el mercado demanda desde el inicio de la "semana cine" (el término utilizado en la industria para definir la convención de unidad de ciclo de exhibición), se evitaría la necesidad de reajustar el cuadro de programación sobre la marcha. De este modo, la participación de mercado aumentaría (aparte de esos puntos porcentuales) pues no se perderían dos días de la semana cine, esperando los cambios del segundo día, con una programación no optimizada.

Por si el margen de mejora considerado pueda resultar en apariencia cuantitativamente limitado, cabe señalar que este apartado ya mostraba un buen desempeño en Cinemark, como lo prueba el hecho de que en porcentaje de ocupación de sala, Cinemark es el líder del sector. A pesar de ello, y con anterioridad al annus horribilis de 2020 para toda la industria, en la compañía ya existía la percepción de un posible desaprovechamiento de las posibilidades de los métodos más recientes de inteligencia de negocio y Big Data en el área de programación.

- # Metodología actual de programación.

En la actualidad, las decisiones de programación en la compañía se realizan de forma tradicional sin optimizar la utilización de los modelos y técnicas analíticas basadas en los datos. Los programadores cuentan con un amplio conocimiento sobre películas y la audiencia de los cines individuales, fruto de años de experiencia. Concluimos que un sistema analítico y predictivo automatizado que complemente el método haría el proceso más eficiente.

Una vez en producción, esto no solo aliviará a los cines de una tarea repetitiva que requiere un considerable esfuerzo e insumo en horas de trabajo, sino que también logrará un mejor desempeño de la programación, medible de forma inmediata en taquilla, que el actual procedimiento.

El ciclo de programación en la industria del cine es semanal. Cada "semana cine" comienza los jueves y se revisa los viernes, para reprogramar en caso de que los resultados no sean los deseados. Por lo tanto, una sala de cine tiene que preparar una nueva programación de películas al comienzo de cada semana y revisarla tras la proyección del jueves. Esto es particularmente complejo para los cines multisala, el formato de cine cada vez más dominante en todo el mundo. La multiplicidad de salas permite más opciones de exhibición pero complica el proceso de decisión.

Para la programación semanal de películas, la gerencia debe determinar qué películas se mostrarán, en qué salas, en qué idioma (V.O.S.E. o doblada) en qué días y a qué horas. Por lo general, en cada sala, un cine puede acomodar de tres a cinco sesiones por día. Esto significa que un cine de diez salas necesita programar alrededor de 280 proyecciones por semana.

• Oportunidades de mejora.

Como hemos visto, las estimaciones de asistencia y programaciones en las salas de cine se basan en la intuición, la experiencia y un análisis relativamente limitado de los datos de facturación iniciales. No existe en este sentido ninguna herramienta de BI en la empresa que permita tomar decisiones en base a un análisis automatizado de los datos o realice predicciones de asistencia utilizando el histórico de desempeño.

La labor ha brindado buenos resultados con el incremento de la experiencia, pero a día de hoy ya no parece posible seguir mejorando por este camino, percepción que Cinemark fundamenta en los siguientes puntos:

- No se pueden conseguir estimaciones estables en el tiempo. Es decir, algunas veces se consiguen estimaciones con un nivel de precisión alto, pero también ocurre lo contrario, con estimaciones muy alejadas de la realidad.
- Se requiere mucho tiempo para analizar la variabilidad de la oferta en las producciones de películas (según género de película, según público objetivo, etc.).
- El método es muy susceptible al error humano, incluso por factores externos al ámbito estrictamente laboral, como pueden ser problemas personales, estrés, cansancio.
- Existe un evidente riesgo de fuga del conocimiento acumulado por salida de la empresa de las personas experimentadas.

- Peca de subjetividad e inconsistencia metódica, resultando en dificultad para el análisis y la trazabilidad de los aciertos o fallos en las decisiones, que imposibilita el aprendizaje y retroalimentación del sistema de toma de decisiones.
- Ausencia de analítica automatizada basada en datos en el proceso de decisión.

Esta falta de optimización en la proyección de la asistencia impacta directamente en un coste de oportunidad por programación, que no solo se traduce directamente en una menor facturación en taquilla, sino que redunda de forma negativa en otras ramas del negocio directamente relacionadas con dicha asistencia y una predicción precisa de la misma, como lo son las siguientes:

- Planificación de las compras y distribución de los alimentos y bebidas, que al no tener una estimación acertada se puede generar altos costos de oportunidad. Los inventarios se abastecen usando las proyecciones de asistencia. Una mejor estimación de las mismas permitirá una gestión más eficiente de stock necesario en sus productos (especialmente de los perecederos) evitando situaciones de exceso o defecto de inventario, e incluso de caducidad de productos.

- Planificación del personal en los cines, que con proyecciones de asistencia inexactas resulta en sobrecostos por una asignación de personal mayor o menor de la necesaria, dando lugar a un mal servicio que afectaría directamente a la satisfacción del cliente y al valor de la marca Cinemark. La estimación de asistencia correcta ayuda a programar los turnos del personal para que la atención al cliente sea la mejor, evitando largas colas en taquilla, optimizando el tiempo y rotación de los servicios de limpieza en sala y baños y asegurando la presencia de empleados de soporte cuando se requiera tal medida. Todas estas mejoras generan un beneficio tangible para la empresa, que incluye la reducción del gasto no eficiente en personal.

Para el departamento de programación, resolver este problema supone una reducción muy considerable de las horas hombre empleadas en esta tarea, que Cinemark cifra a día de hoy en el 20% del tiempo total de la semana del departamento, por lo que el impacto de una mejora en estos procesos se traduce en un beneficio significativo.

• ¿Qué aspectos importantes se conocen sobre el problema que pudieran contribuir a su solución?

La industria del cine es transparente con relación a los datos de asistencia e ingresos, existe una empresa llamada Comscore que con su herramienta IBOE centraliza la información que le reportan todos los cines de todas las cadenas de la región. En este repositorio de datos existe información de la Industria de más de 15 años, histórico que ya se usa para tomar decisiones en la empresa actualmente, por lo que contar con esta fuente de datos (así como con la información interna de Cinemark) es el aspecto más importante que puede contribuir a una solución, pues a través de su integración en una herramienta de aprendizaje automático y procesamiento de lenguaje natural se pueden detectar patrones que no son visibles usando las técnicas tradicionales.

Validación de hipótesis.

Identificación de hipótesis a validar.

En base a la problemática anteriormente expuesta de las programaciones manuales y el impacto que esta metodología pueda tener en la facturación de Cinemark nos planteamos las siguientes hipótesis:

Hipótesis:

1. Existe margen de mejora en las estimaciones de facturación que hace Cinemark.
2. Las decisiones de programación inciden de forma determinante en la facturación y asistencia a las salas.
3. De los factores que determinan dichos niveles de asistencia y facturación en las salas, (productora, elenco, director, género, formato de pantalla, hora y día de exhibición) es la programación aquél en el cual podemos influir de forma más directa y evidente.
4. Basando las decisiones en datos, asumimos una mejora en la facturación.
5. Podemos considerar un éxito la programación cuando iguala o mejora las cuotas obtenidas por los competidores en el mercado regional en las cinco películas más taquilleras.
6. La mejora en la estimación de los asistentes redunda en una mejor planificación en las actividades de negocio paralelas (abastecimiento de inventario, gestión de personal, etc.).
7. Con nuestra herramienta, el tiempo dedicado a la actividad de programación se verá reducido significativamente.
8. Aprovechando la información histórica de los últimos cinco años, disminuirán en número y frecuencia las reprogramaciones.
9. La programación es un factor que afecta a la elección del cine.
10. Los datos de Comscore son una fuente de datos apropiada, disponible y fiable para alimentar el modelo y mejorar las predicciones.
11. Será conveniente incorporar otras fuentes de datos para completar la información, en especial con vistas al desarrollo de la utilidad de PLN.

Hipótesis descriptivas (azul), correlacionales (roja) y exploratorias (morado).

- # Proceso de validación (entrevistas y encuestas)

De cara a contrastar las hipótesis planteadas hemos recurrido a diversas fuentes, tanto documentales como presenciales.
En primer lugar hemos tenido una entrevista con Fernando Collado, Head Film Buyer de Cinemark CAM. Igualmente hemos sondeado a los jefes de programación de Cinemark en América Latina, cuyas impresiones coinciden con las de la filial de Centroamérica. Por su importancia, transcribimos la entrevista a continuación.

<u>Detalle de la entrevista con Fernando Collado, Head Film Buyer de Cinemark CAM:</u>

1. ¿Has usado alguna vez los datos para tomar decisiones?
"Sí, tanto las estimaciones de asistencia como la programación se hacen en base a datos históricos que se encuentran en la web de Comscore y en las base de datos interna de Cinemark. Un punto importante sobre la programación es que aparte de datos históricos se utilizan los datos reales que se generan la semana anterior, para saber qué películas deben mantenerse en cartelera en la siguiente semana y cuáles deben salir."

2. ¿Recuerdas alguna ocasión en la que te hubiera gustado tener más datos para decidir la programación?

"Sí, en la mayoría de semanas llegan películas medianas o pequeñas de las que no se tiene mayor información, el comparativo es difícil de realizar y al final la estimación de asistencia se hace sin tener una base sólida, lo que afecta también a la programación, pues no se tiene certeza de si se está dando un espacio de cartelera mayor o menor del que realmente necesita o merece."

3. ¿Qué margen de mejora tienes en la programación?
"*Es un margen amplio, se pueden mejorar tanto los horarios de exhibición de las películas como la elección del idioma más conveniente, para ambas decisiones se debe considerar el cine donde se está programando, pues no todos los cines son iguales. Considero que establecer criterios claros para cada uno de los cines es importante y sería una mejora, no todos los géneros de películas funcionan igual en cines de distintos países o siquiera en la misma ciudad, este me parece es uno de los principales aspectos de mejora. Actualmente la base de la programación es la misma para todos los cines, luego se ajusta para cada uno de acuerdo a los criterios del programador, pero este ajuste viene dado siempre por experiencia y no por datos."*

4. ¿Alguna vez has medido el incremento de la facturación derivado de un ajuste en la programación?
"*Esa medición se hace todas las semanas en las que hay un ajuste, el día viernes por la mañana se mide la participación de mercado de la empresa resultante de la programación original. Con base a los resultados que se vean ese día se hacen los ajustes de programación que se consideren necesarios y el día lunes se mide el efecto que tuvieron los ajustes. En promedio, una vez se hacen las correcciones oportunas la facturación del fin de semana sube un aproximado de 2% y 3%."*

5. ¿Recuerdas un caso reciente en el que la programación hizo caer gravemente la facturación?
"*Sí, la película Bad Boys en la cuarta semana de Enero 2020 se programó muy conservadoramente y la película Doolittle se programó muy agresivamente. El día de estreno nos llevamos la sorpresa de que tuvimos en general una participación de mercado 4% menor que lo que teníamos como tendencia para nuestros días de estreno, esto se debió a que Bad Boys fue la película número 1 por mucho en ese día, generando 4 veces más de lo que generó Doolittle."*

6. ¿Crees que se podría haber evitado? ¿Cómo?
"*Es claro que se pudo evitar, se debió haber programado al revés, ser agresivo con la película que resultó ser la #1 y más conservador con la #2. Los ajustes a la programación llevaron el resto de días del fin de semana de estreno a niveles de ingreso normales."*

7. ¿Cómo crees que afecta la programación en la facturación?
"*Afecta en gran medida, los vemos semana tras semana, una programación que tenga variedad y que oferte lo que los clientes quieren ver es clave. Las redes sociales nos han abierto los ojos en este sentido, vemos quejas sobre películas que en algunos cines se programan únicamente en un idioma o muy temprano o muy tarde. Esto, anteriormente, tan solo lo intuíamos; ahora lo sabemos de primera mano y notamos su importancia."*

8. ¿Qué factores crees que determinan los niveles de asistencia?
"*Aparte de la buena programación, hay otros factores que también inciden en la asistencia como la promoción a las películas y el esfuerzo que se haga en mercadeo para la cadena de cine, así como también la calidad de los alimentos y bebidas que vendemos, pero en líneas generales la programación es el factor que más afecta a la facturación de una cadena de cine."*

9. ¿Cuál sería la influencia de una buena programación sobre la asistencia del público?
"Es la principal influencia, una buena programación atrae mayor asistencia, incluso cuando no hay alguna película fuerte en cartelera siempre se tiene la comparación con la competencia, la cadena de cine que programe mejor el poco contenido que se tenga disponible para una semana de temporada baja, por ejemplo, es el que obtiene mayor participación de mercado."

10. ¿Qué haría cambiar de plan a tu cliente y elegir otro cine?
"No encontrar la película que desean ver o encontrar que sí está programada la película que desea ver, pero está en una hora que no le conviene o en un idioma que no le gusta, son las casos que nuestros taquilleros nos comentan repetitivamente, en los que una persona que ya está en el cine lista para comprar su boleto, les da las gracias y finalmente no ingresa."

11. Mejorar la estimación de la asistencia basada en la información histórica de 2 años, ¿nos permitirá reducir las reprogramaciones?
"Sin duda, la idea de las reprogramaciones es colocar funciones extra en las películas que tienen un rendimiento mayor al que se esperaba y quitarle funciones a las películas que no llenan las expectativas. Si la programación original ya contempla que una película no tendrá un desempeño aceptable entonces no entraría en cartelera y por el contrario se daría más funciones a los títulos 1 y 2 por estimación de asistentes, en este caso no habría necesidad de efectuar una reprogramación."

12. De automatizar la programación, ¿cuánto tiempo se reduciría?
"Es claro que se reduciría considerablemente, De momento somos 2 personas que utilizan 2 días de su semana completos, yo estimo una reducción de hast un 70% en este tiempo al usar una herramienta que automatice."

13. ¿Crees que tus estimaciones son acertadas?
"En general, el error en estimaciones es de un +/- 15%, es algo suficientemente bueno y se puede trabajar con ello. Ese margen de error es la principal razón de las reprogramaciones."

14. ¿Dedican muchos recursos en la empresa a la programación/reprogramación semanalmente?
"2 personas dedican el 20% de su semana a estas tareas."

15. ¿Son necesarias diferentes fuentes de datos para llevar a cabo la programación?
"Sí, la riqueza que da tener los comparativos que hacer Cinemark USA, la información con la que contamos internamente en Cinemark Centroamérica y la proveniente de Comscore son todas herramientas fundamentales para poder generar una buena programación, sin esos datos estaríamos en una situación en la que tendríamos que inventar programaciones y esto aumentaría los errores."

16. ¿Mejorar la estimación de la asistencia de las películas nos ayuda en la mejora de la planificación de los alimentos y personal?
"Sí, tanto la planificación de compra de alimentos como la planificación del personal que debe trabajar en el cine cada fin de semana es finalmente calculado en base a la estimación de asistencia que realicemos. Recibimos consultas constantes de los encargados de ambas áreas precisamente por esto, ellos tienen sus propios indicadores que deben mantener en meta por lo que tener buena información de nuestra parte es crítico para ellos."

También nos hemos puesto en contacto con varios exhibidores en España, y en ningún caso han confirmado la utilización de herramientas automatizadas en su proceso de programación; de hecho, algunos han confirmado su no utilización. Sin embargo, tenemos constancia de la existencia de soluciones similares a la que pretendemos construir, tanto de predicción de asistencia a las salas como de optimización de la programación, por parte de exhibidoras en los ámbitos europeo y norteamericano. No nos ha sido posible, sin embargo, obtener datos concretos del desempeño de estas herramientas o su efectividad respecto a años anteriores a su uso.

Cabe mencionar asimismo un trabajo de 2009, "*Demand-Driven Scheduling of Movies in a Multiplex*" (Eliashberg, Hegie, Ho, Huisman, Miller, Swami, Weinberg, Wierenga), *en* el que ya se estudia la necesidad de apoyar decisiones de programación en algoritmos basados en datos. Entre otras cosas, se argumenta la tesis *"El pronóstico y la programación de películas en la práctica tienden a asociarse con la intuición más que con el análisis y esto también caracteriza la tradición de la toma de decisiones en este dominio"*.

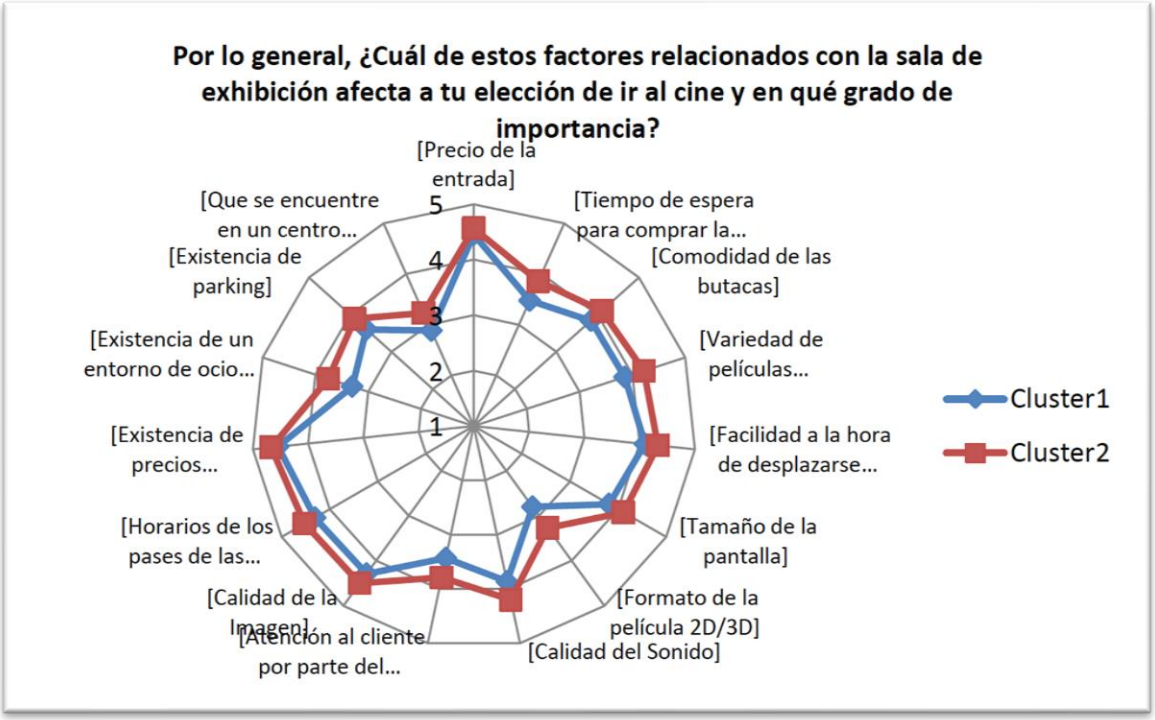
En otro estudio consultado en la etapa de investigación y documentación preliminar (cfr. Bibliografía) Por último, entre otras cuestiones, se analizan los comportamientos del espectador de cine expresados en una encuesta a 5.314 personas. En ella se formula la pregunta

¿cuáles son los factores relacionados con las salas de exhibición que afectan la elección de ir al cine y en qué grado de importancia?

Y se permite responder *Mucho/Bastante/Ni mucho ni poco/Poco/Nada* en los siguientes apartados:

1. Precio de la entrada
2. Tiempo de de espera para comprar la entrada
3. Comodidad de las butacas
4. Variedad de películas proyectadas en el cine
5. Facilidad a la hora de desplazarse al cine
6. Tamaño de la pantalla
7. Formato 2D/3D
8. Calidad del sonido
9. Atención al cliente por parte del personal de la sala
10. Calidad de la imagen
11. Horarios de los pases de las películas
12. Existencia de precios promocionales reducidos
13. Existencia de un entorno de ocio próximo
14. Disponibilidad de zona de aparcamiento
15. Que se encuentre en un centro comercial

Debe mencionarse que esta encuesta se ha realizado en España, pero consideramos que las respuestas son razonablemente extrapolables, siempre con un carácter puramente preliminar, comportamiento del espectador en la región de ulterior aplicación de nuestra herramienta.



Gráfica obtenida de Gil Martín, Mª Montserrat (2018) *Comportamiento del consumidor de cine en salas: factores motivacionales y tipología del consumidor.*

Evidencia de la programación manual o uso de algoritmos basados en datos:

1. A la pregunta formulada: ¿se está utilizando una herramienta de este tipo en las salas españolas, o si las programaciones se realizan y deciden de forma exclusivamente tradicional, sin ayuda de algoritmos basados en datos?

"*Nosotros lo hacemos todo de manera manual e instintiva, sin algoritmos. Muy de andar por casa.*"

Responsable de márketing de una cadena de cines (Madrid).

2. En la página web de la empresa Cinema Intelligence y la de Vista, que se dedica a entre otras cosas vender una solución similar a la que planteamos, aparecen testimonios de empresas que han trabajado con ellos:

"*En Pathé pasamos por el proceso de centralizar la programación para 201 pantallas. Toda la planificación de horarios y películas está centralizada, lo cual nos facilita el proceso por la mayor facilidad de control y el ahorro de tiempo. Estas mejoras han sido posibles gracias a Cinema Intelligence. A través del análisis de los datos de que disponemos y del aprendizaje*

resultante del entrenamiento de su herramienta con dichos datos, Cinema Intelligence puede crear una tabla de horarios optimizados para cada día, lo que nos ayuda mucho.Nosotros, mi equipo y yo, como expertos en programación, siempre tenemos el control del resultado final.”

Daniella Koot, Directora de programación de Pathé en Países Bajos.

“*Cinema Intelligence es un producto verdaderamente visionario.Con Cinema Intelligence, Cinemaxx ahora puede maximizar las oportunidades del nuevo mundo del cine digital.”*

Steffen Schier, jefe de programación en Cinemaxx, Alemania.

“*Estábamos buscando soluciones de software que pudieran incorporar análisis de datos y proporcionar información procesable sobre nuestro negocio. Estamos encantados de asociarnos con Cinema Intelligence. Después de valorar la potencialidades y los posibles beneficios, concluimos que el software ofrece una enorme oportunidad de ahorro de tiempo y dinero en nuestro proceso de programación. El equipo está muy ilusionado.”*

Brock Bagby, vicepresidente de programación y desarrollo de negocio B&B Theatres, EEUU.

“*El área de datos era una fuente constante de problemas en Bow Tie, especialmente en relación con la tecnología, lo que suponía un grave obstáculo para el correcto desempeño del negocio. Para resolver estas dificultades, decidimos recurrir a Vista.”*

Bow Tie Cinema, EEUU.

• Análisis y conclusiones derivadas de las entrevistas y encuestas

Los principales problemas que la empresa nos ha indicado que tienen actualmente son:

- Pérdida de participación de mercado respecto a la competencia.
- Datos de predicción errados que se envían a otros departamentos.
- Asignación de turnos a los empleados ineficiente.
- Problemas con el manejo de inventario de alimentos o bebidas.

La empresa responde con interés al planteamiento de una herramienta de soporte que permita automatizar los pronósticos de asistencia y nos indican su total apoyo con cualquier consulta que el grupo tenga.

En base a las respuestas obtenidas en la entrevista, podemos cuantificar algunas de las hipótesis que se han visto validadas.

- *"En promedio, una vez se hacen las correcciones oportunas la facturación del fin de semana sube un aproximado de 2% y 3%,"* pregunta 4.

Nos indica que podemos esperar una mejora en la facturación de un 2-3%, es decir situarnos en una cuota de mercado para Cinemark del 23-24%, si somos capaces de generar programaciones ajustadas. Afecta a las hipótesis 2 y 3.

H2*. Las decisiones de programación inciden de forma determinante en la facturación y asistencia a las salas.
H4*. Basando las decisiones en datos, la facturación aumentaría entre 2 y 3%.

- "*De momento somos 2 personas que utilizan 2 días de su semana completos, yo consideraría una reducción hasta de un 70% en este tiempo al usar una herramienta que automatice*", pregunta 12.

Esto afectaría a la hipótesis 7.
H7*. Con nuestra herramienta, el tiempo dedicado a la actividad de programación se verá reducido significativamente (un 70%).

- *"El error en estimaciones es de un +/- 15%, es algo suficientemente bueno y se puede trabajar con ello. Ese margen de error es la principal razón de las reprogramaciones.",* pregunta 13.

En base a esta respuesta podríamos modificar la hipótesis 1 para incorporar la información cuantitativa que se aporta.
H1*. Existe margen de mejora en las estimaciones de facturación que hace Cinemark -> El 75% de las estimaciones en la facturación que hace Cinemark se alejan más de un 5% del monto final.

Además, la respuesta a la pregunta 7 "*siempre llegan películas medianas o pequeñas de las que no se tiene mayor información, el comparativo es difícil de realizar y la estimación de asistencia se hace sin tener una base sólida y esto afecta también a la programación, pues no se tiene certeza de si se está dando un espacio de cartelera mayor o menor del que realmente necesita o merece*" induce a pensar que puede ser interesante para la compañía contar con una herramienta que permita comparar estrenos con películas nuevas, a partir de datos básicos como sinopsis, elenco, director o título. En este sentido, habría que modificar ligeramente la hipótesis 11 para incluir esta problemática.

H10*. Los datos de Comscore son suficientes para alimentar el modelo (y obtener una predicción fiable) en los grandes estrenos.

H11*. Para estimar películas medianas o pequeñas, hace falta complementar el modelo con un comparador de películas, que puede entrenarse con datos públicos de IMBD, Rotten Tomatoes, etc.

Hemos encontrado una afirmación a nuestra hipótesis "*De los factores que determinan los niveles de asistencia y por tanto de facturación de las salas, el más influyente y en el cual podemos influir*

es la programación" en las respuestas a la pregunta que se muestra en la gráfica de la página 14, es importante destacar que entre las opciones dadas a los encuestados que tienen relación directa con las tareas del Departamento de Programación el punto "*Horario de los pases de las películas*" ha sido el más seleccionado.

Finalmente, en base a las comunicaciones con exhibidores en España y a los testimonios que hemos encontrado en otras compañías, podemos afirmar que el uso de herramientas de Inteligencia Artificial no está ampliamente extendido en este sector, sobre todo en los exhibidores más pequeños, y detectamos un interés de parte de los participantes del mercado en una solución de este tipo.

Análisis preliminar de datos

• ¿De qué datos disponemos?
Disponemos de los datos internos de Cinemark así como todos los datos históricos de la industria de los últimos veinte años en Comscore, de los cuales usaremos los de un número de años a determinar, pues no se consideran extrapolables datos de hace tanto tiempo, dadas las transformaciones que ha sufrido la industria.

• ¿Qué datos a los que tenemos acceso no se están recogiendo?
Los pósters o cartelería de los estrenos. Ahora mismo no se recoge esta información, pero sería posible utilizar esta información de cara a estudiar relaciones entre películas.

• ¿Qué datos pueden generarse a partir de nuestros productos y operaciones?
En primer lugar, vamos a generar comparativas de películas, para generar estimaciones de asistencia y así poder crear las programaciones sugeridas. Además, las propias estimaciones de asistencia, darán lugar a una colección que nos permitirá evaluar el modelo según se cumplan o no dichas estimaciones.

• ¿Qué datos podríamos obtener de otros que nos serían de utilidad?
Existen datos públicos sobre producciones cinematográficas, como IMDB o Rotten Tomatoes, que cuentan con opiniones de público y crítica, además de sinopsis, información sobre elenco, equipo, etc. Estos datos nos pueden ser de gran ayuda de cara a relacionar estrenos (de los que no se conocen datos de cartelera aún) con películas pasadas.

Datos demográficos de las zonas de influencia en las que se encuentra cada cine (edad, población, ingresos, etc.).

• ¿Qué datos que tienen otros podríamos usar en una iniciativa conjunta?
La industria actualmente ya vuelca diariamente todos los datos en un repositorio común (Comscore).

• **¿Cómo podríamos estructurar y analizar nuestros datos para generar mayor valor?**
En primer lugar, nos planteamos descargar todos los datos históricos de Comscore de los últimos cinco años, así como los datos de Cinemark que complementan esta información (en particular desgranando los horarios por género). Después de llevar a cabo el proceso de extracción, transformación y carga (ETL), obtendremos una base de datos que contenga toda la información que nos interesa. Aparte, vamos a crear una base de datos que nos permita comparar películas y encontrar similitudes entre las mismas, teniendo en cuenta aspectos como: título, sinopsis, director, actores, pósters, etc. La idea es explotar la potencia del PLN en las variables de texto, y ML con las imágenes.

• **¿Estos datos son valiosos internamente para nosotros, o para nuestros clientes actuales, o para nuevos clientes potenciales, o para otras industrias?**
Los datos, tanto los que ya se tienen como los que vamos a generar, son valiosos internamente para nosotros y para nuestros clientes actuales (Cinemark AC en nuestro caso). También serían de interés para otros clientes del sector, aunque claramente haya conflicto de intereses al utilizar los datos propios de Cinemark. Pero la solución creada sería extrapolable a otras empresas, utilizando sus datos propios.
En el caso de otras industrias del entretenimiento presencial, como el teatro, o la televisión (en el predictor de audiencias), podría aplicarse, con las modificaciones necesarias para adaptar la herramienta a la especificidad de dichos comparables. Para industrias no comparables, no sería de interés por tratar un problema muy específico de la actividad exhibidora.

Análisis del entorno.

A la hora de analizar el entorno en el que se desarrollará el proyecto, hemos de tener en cuenta distintos factores, que exponemos a continuación.

• **Factores político-jurídicos.**
Las restricciones de aforo debidas al Covid-19, que diferirán de país a país, son un hecho sin precedente que no recoge el histórico de los datos.

• **Factores culturales.**
En la región de Centro América la asistencia al cine está muy instaurada en la sociedad, tratándose de un sector en alza dentro de los planes de ocio. También el cine supone una opción que reporta seguridad a los espectadores, por hallarse en zonas alejadas de focos de violencia.

• **Factores económicos**.
Marco de crisis mundial por el impacto de la pandemia, especialmente acusado en la industria exhibidora,
Es posible que entre las medidas económicas que se tomen para salir de la crisis en la región se incluyan subidas de impuestos que afectarían claramente al sector. Además, el cine, al no ser un artículo de primera necesidad, se verá afectado por una bajada de asistencia.

• **Factores socio-demográficos.**

En la región de América Latina, la población es muy joven (media de 25 años) con una natalidad alta (19 nacimientos/1000 habitantes). Actualmente la pirámide poblacional parece estable. No se espera a corto-medio plazo movilidad de la población de zonas rurales a ciudades, ni viceversa.

• **Factores tecnológicos.**
A día de hoy, existen tecnologías que permiten "replicar" la experiencia de cine desde casa, con la mejora y a la vez abaratamiento de las televisiones digitales, proyectores, equipos de sonido, etc. Aunque no son accesibles al grueso de la población, sí pueden suponer una diferencia en zonas económicamente más favorecidas.

• **Factores medioambientales.**
Basándonos en la experiencia de décadas, no se identifican factores medioambientales que afecten significativamente a la asistencia a salas de cine en la región de aplicación del proyecto. El clima es poco cambiante y bastante estable, sin fenómenos meteorológicos extremos que supongan interrupción en la actividad comercial.

Análisis competitivo.

Entender cuáles son los competidores y qué propuesta de valor ofrecen es importante para diferenciar la nuestra de lo que actualmente existe en el mercado y estudiar qué otras soluciones de Big Data se han implantado en las empresas competidoras de nuestro cliente.

Hemos identificado a través de consultas que un número importante de cines y cadenas de cine internacional no utiliza una solución basada en Big Data para el problema en el que nos estamos centrando. Y constatamos que las programaciones se siguen haciendo "manualmente" en la mayoría de los casos.

También hemos encontrado empresas que ofrecen soluciones similares a las que deseamos crear y que operan a nivel global, aunque aún no hay presencia en el mercado Centroamericano. La anteriormente citada Cinema Intelligence, comenzó a construir soluciones BI para cine en 2009 en Holanda pero no presentó su producto actual hasta 2014. Otro actor es Vista Entertainment Solutions, una empresa neozelandesa con presencia global que comenzó su andadura en 1996 y que cuenta entre otras soluciones con un planificador de programación similar al que queremos desarrollar.

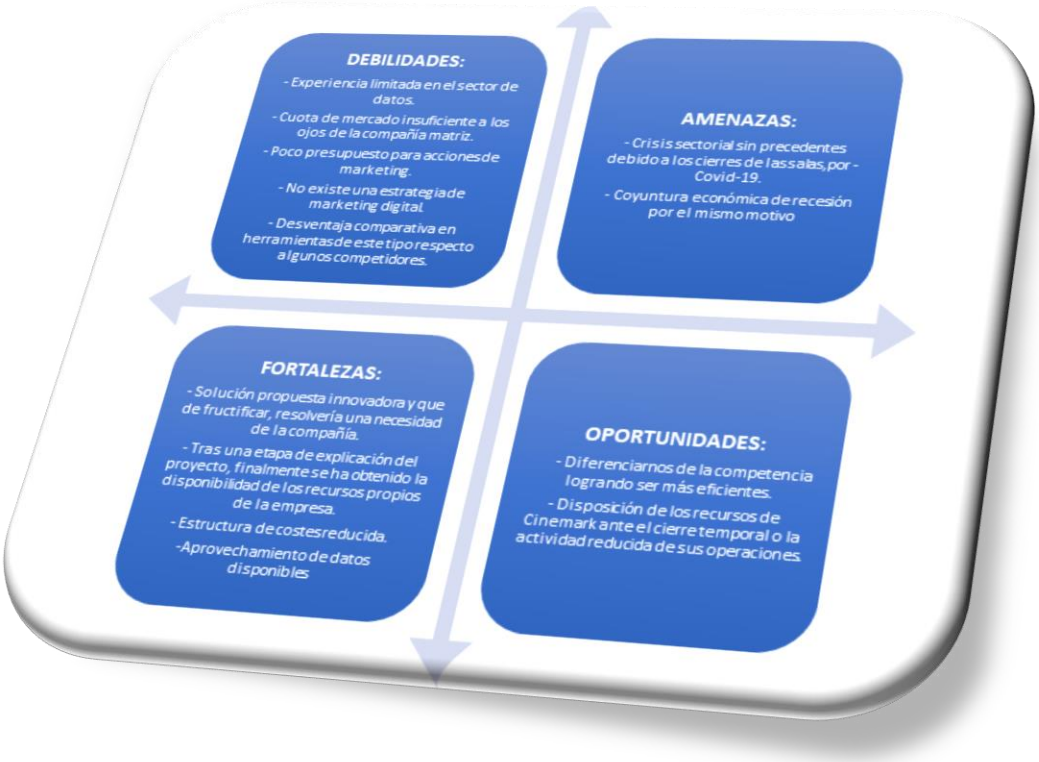
En el mercado regional los competidores de Cinemark no cuentan con una solución de Big Data. Conocemos que otra filial de Cinemark (Brasil) tiene implementada una solución de PowerBI para reporting, pero no se utiliza para diseñar o sugerir programaciones.

En ese sentido, se puede concluir que la demanda en el mercado de soluciones de este estilo existe y, aunque existen competidores que ya son capaces de implementar productos similares, en la región centroamericana sería algo nuevo y para Cinemark supondría una clara diferenciación en el mercado.

Análisis y diagnóstico.

El origen del proyecto reside en la percepción que la división centroamericana de Cinemark tiene de la necesidad de crear una herramienta para la predicción del desempeño de las películas en su exhibición. Esta solución ayudaría a mejorar su proceso de decisión de programación, que en este momento la compañía considera no está optimizado, dado que se realiza de forma intuitiva y no predominantemente basado en datos.

Análisis DAFO.



Debilidades:

- Experiencia limitada en el sector de datos.
- Cuota de mercado insuficiente a los ojos de la compañía matriz.
- Bajo presupuesto para acciones de marketing.
- Inexistencia de una estrategia de marketing digital.
- Cultura corporativa que muestra cierta renuencia a la adopción de nuevas soluciones.
- Desventaja comparativa en herramientas de este tipo respecto a algunos competidores (si bien no en la región de Centro América).

Amenazas:

- Crisis sectorial sin precedentes debido a los cierres de las salas por Covid-19.
- Coyuntura económica de recesión por el mismo motivo.

Fortalezas:

- Solución propuesta innovadora y que de fructificar, resolvería una necesidad de la compañía.
- Tras una etapa de explicación del proyecto, finalmente se ha obtenido la disponibilidad de los recursos propios de la empresa.
- Estructura de costes reducida.
- Aprovechamiento de los datos disponibles.

Oportunidades:

- Diferenciación de la competencia logrando ser más eficientes.
- Disposición de los recursos de Cinemark ante el cierre temporal o la actividad reducida de sus operaciones.

Definición del modelo de negocio.

Socios clave:

- Cinemark, cadena de salas de cine propiedad de Cinemark Holdings, Inc., con 534 salas y 5.977 pantallas en 16 países.

Recursos clave:

Intelectuales:

- Datos de Comscore (www.iboe.com)
- Información externa sobre sinopsis, elencos, etc. (IMDB, OMDB)
- CMU Movie Summary Corpus: Corpus de texto con los diálogos de más de 6000 películas que mediante técnicas de PLN será utilizado en el modelo de predicción.

Actividades clave:

- Identificación de las variables objetivo del modelo de predicción.
- Identificación de las variables independientes disponibles para realizar la predicción.
- Recopilación de los datos de aquellas variables no inmediatamente disponibles en las fuentes de datos iniciales mencionadas.
- Extracción, transformación, limpieza y carga de los datos.
- Análisis exploratorio de los datos.
- Creación, optimización y entrenamiento del modelo.
- Desarrollo de la herramienta de visualización y su integración en la interfaz.
- Generación de la interfaz de interacción con el cliente.
- Mantenimiento y actualizaciones de ambas herramientas.

Propuestas de valor:

- Aumento de la cuota de mercado de la empresa, optimizando los espacios en pantalla para los títulos más rentables.
- Optimización de otros aspectos de la gestión, como restauración, personal, consumo energético.
- Reducción del tiempo horas hombre invertido en la tarea semanal de programación.
- Optimización del abanico de opciones que se ofrece al cliente, incrementando la satisfacción y mejorando la percepción que de la empresa tienen los espectadores.
- La proyección que el modelo genera mejora la actual en varios aspectos: exactitud, fiabilidad, cuota de mercado, niveles de granularidad (sala, cine, país).
- Creación de un repositorio de datos tanto reales como de sus predicciones. Esta información actualmente no es almacenada por Cinemark y no se lleva a cabo análisis ni existe un historial de desempeño de predicciones pasadas.
- El análisis de los datos provenientes de esta solución permitirá mejorar el conocimiento de los clientes, pudiendo adoptar decisiones específicas para cada emplazamiento de negocio.
- Sociedad commercial.
- Fidelización del cliente a través de una programación más adecuada a los patrones de comportamiento.

Canales:

- Reuniones periódicas de seguimiento.
- La interfaz web.
- Los cuadros de mando de Tableau.

Segmentos de clientes:

- Salas de cine exhibidoras (Cinemark).
- Los demás departamentos de Cinemark locales y regionales (Logística, Recursos Humanos, Finanzas).

Estructura de costes:

- Personal.
- Equipos informáticos.
- Servicios en la nube (bases de datos, Azure, web hosting, dominio).
- Licencias (Comscore, Tableau Server).

Fuentes de ingresos/beneficios:

- Relación contractual con nuestro cliente (Cinemark).
- Consultoría.
- Mantenimiento y actualización:
- Las bases de datos se actualizan con los nuevos registros de las películas que van completando su ciclo de exhibición; estos nuevos datos requieren tratamiento.
- Reentrenamiento periódico y afinación/corrección del modelo incorporando los nuevos datos.
- Seguimiento de la evolución del desempeño de la función predictiva.

Plan de acción.

Objetivos.

- El objetivo es el aumento de la cuota de mercado de Cinemark mediante la creación e implantación de una herramienta de predicción del comportamiento del mercado, a través de la estimación del desempeño de las películas durante su ciclo de exhibición y de la optimización de la programación.

- Sobre las predicciones resultantes y el análisis del modelo se fundamentarán las decisiones de programación, ubicación, marketing, teniendo como referencia de desempeño el aumento en la cuota de mercado de Cinemark.

- La predicción se basará en los datos obtenidos respecto al desempeño de las diferentes producciones cinematográficas en la industria exhibidora, tanto a nivel agregado como por salas de exhibición, y será calculada a través de un modelo algorítmico de predicción en aprendizaje automático.

- La implementación de la herramienta supone una evolución en el proceso de toma de decisiones de programación, uno de los aspectos esenciales en el lado del negocio de Cinemark.

El carácter dinámico del mercado de la exhibición determina que la herramienta predictiva recoja dicho dinamismo, para lo cual requerirá de seguimiento y actualizaciones continuas a través de análisis de desempeño predictivo, y reentrenamientos, pruebas y revisiones periódicas.

¿Cuándo? El estimador debe estar listo para entrar en producción el 1 de noviembre de 2020. El programador de salas, horarios y otros aspectos de la exhibición (segunda fase del proyecto), el 1 de febrero de 2021.

¿Dónde? La zona de aplicación inicial de la herramienta será Centroamérica, en la que Cinemark cuenta con salas en Costa Rica, El Salvador, Guatemala, Honduras, Nicaragua y Panamá.

El desarrollo de la herramienta se realiza de forma colaborativa entre El Salvador, España, y Perú. Su implementación se llevará a cabo en la oficina central de El Salvador, ubicación de la central de nuestro cliente Cinemark para su división de Centroamérica.

Métricas.

Los indicadores principales que permitirán evaluar la obtención de los objetivos estratégicos son principalmente los siguientes:

- El aumento de la Participación de Mercado en Centro América: Este indicador está directamente relacionado al objetivo estratégico de Cinemark, el cual empieza con una visión de toda la Región Centroamericana y luego se va revisando por País, Género de Películas y por Películas.

- El aumento de las Asistencia de Clientes durante la primera semana de estreno. Este indicador permite medir la capacidad instalada de las salas de cine. Este indicador se puede analizar con varios cruces de otras variables, debido que se incremento puede deberse a inauguraciones de nuevas salas de Cine, películas más taquilleras o esperadas, etc.

- Ventas por Asistencia de Clientes. Correspondientes exclusivamente a las ventas de entradas de Cine, sin considerar otros ingresos como ventas de bebidas y alimentos, eventos corporativos, etc.

- Porcentaje de Acierto entre Asistencia Real vs. Predicción de Asistencia. Este indicador permitirá ver cuándo el modelo necesita ser reentrenado, incorporando las películas más recientes.

- Indicadores de Medición de Desempeño de los Modelos. Estos indicadores permitirán evaluar la precisión y exactitud del modelo con el set de Datos de Test, previo a su puesta en producción, esto para cada ciclo de reentrenamiento.

SITUACIÓN GENERAL DE LA INDUSTRIA EN CENTROAMERICA

Market Share	Asistencia de FdS	Asistencia FdS Cinemark	Ventas FdS Centroamérica	Ventas FdS Cinemark
29.56%	449,170	132,783	2,414,784	662,253

Una mejora en la facturación de un 2-3%, es decir, situarnos en una cuota de mercado para Cinemark del 23-24%.

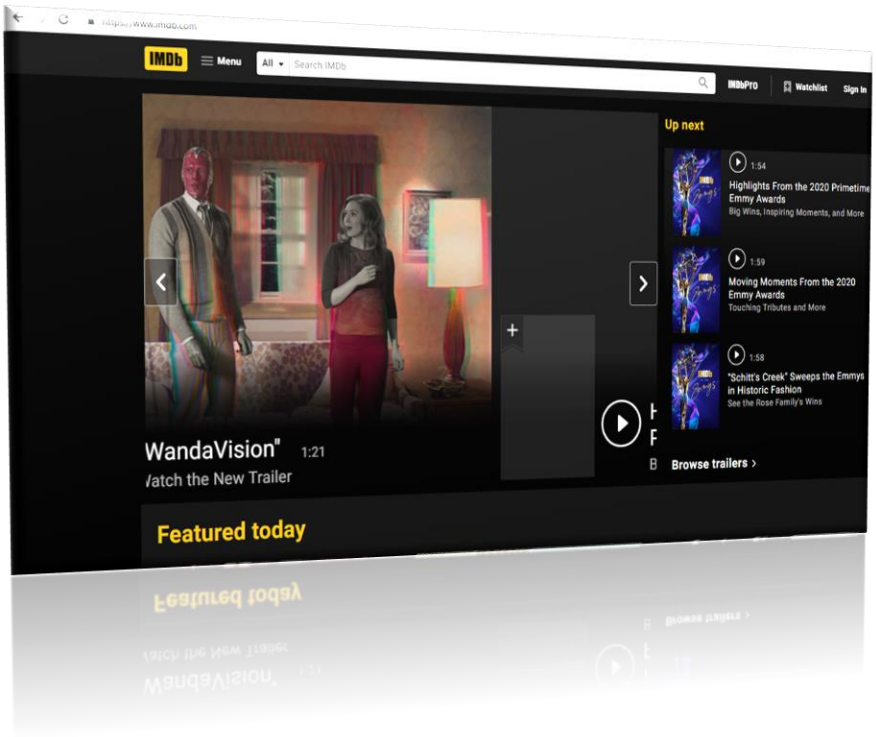
Tareas.

Identificación de los datos.

• Bases de datos de terceros (públicas o privadas)

La mayor parte de los datos los extraemos de Comscore, que es un repositorio de toda la información de los cines de todo el sector. La industria del cine comparte sus resultados diariamente entre competidores a través de un portal web, por lo que saber la posición de una empresa con relación al mercado es fácilmente verificable por semanas, días o el rango de fecha que se quiera analizar, por tanto, esta fuente de datos externa está garantizadas y disponible 24/7.

La base de datos pública que utilizaremos será la información que se almacena en IMDB y que se accederá a través del paquete IMDBPy (https://imdbpy.github.io/).

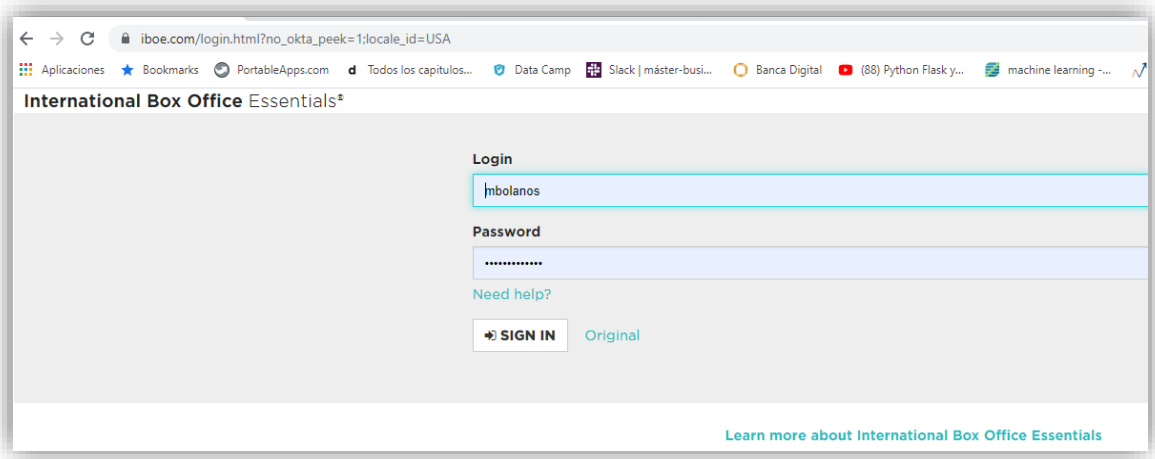


IMDBPy es un paquete Python que nos permite obtener información directamente desde IMDB en Python con comandos simples, se pasa la información a dataframes de Python que hacen match con las películas obtenidas desde Comscore para tener información de entrenamiento completa.

Otros datos se obtienen directamente de Google a través de scripts creados específicamente para ello. Información como el presupuesto de la película, los actores, directores, sinopsis, etc, complementan la información de Comscore y nos permite alimentar nuestro modelo con datos de mayor calidad.

Captura de datos.

- La fuente principal de datos que alimentará nuestra herramienta es el sitio web de Comscore. Ante la imposibilidad de obtener acceso a la API de Comscore, se procedió a crear una rutina automatizada para la obtención de los datos.

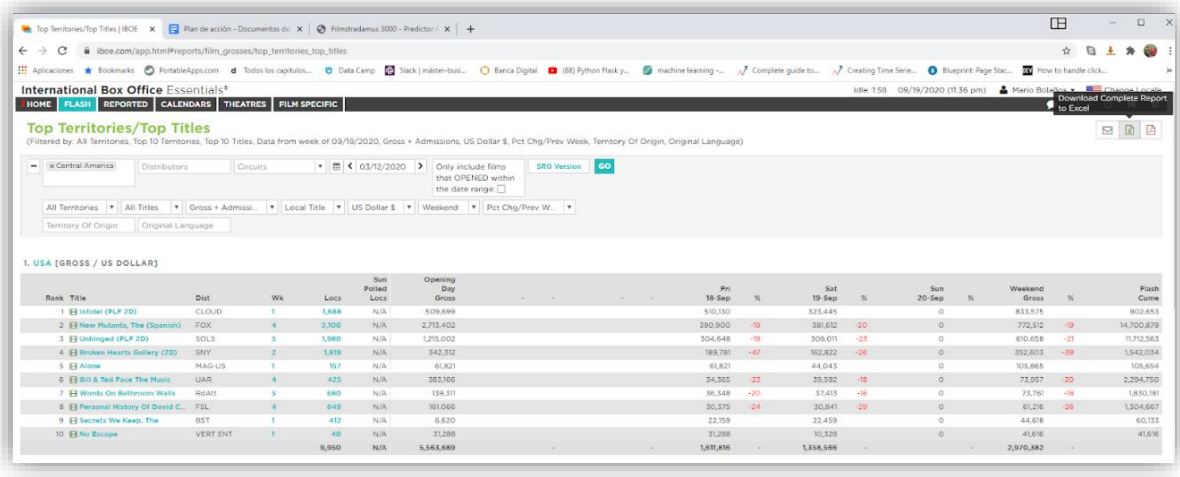


Acceso a la web de Comscore.

- La rutina utiliza Selenium con Python interactuando con el frontend de la web de Comscore en www.iboe.com para poder descargar toda la información deseada.

- La información se descarga en archivos que se vuelcan en su estado original a dataframes Python, pasando directamente al proceso de transformación y validación.

- Es importante mencionar que Comscore no detalla en ninguno de sus reportes información general de las películas, Director, elenco, Presupuesto o Sinopsis. Es por este motivo que se ha hecho necesario el uso de la API de OMDB e IMDB para recabar esos datos. Para aquellas películas que aún no se encuentran en esos sitios o de las cuales no se incluye esta información, se desarrolló una rutina de web scraping dirigida directamente a Google como último recurso.



Captura previa a la descarga de datos en Comscore.

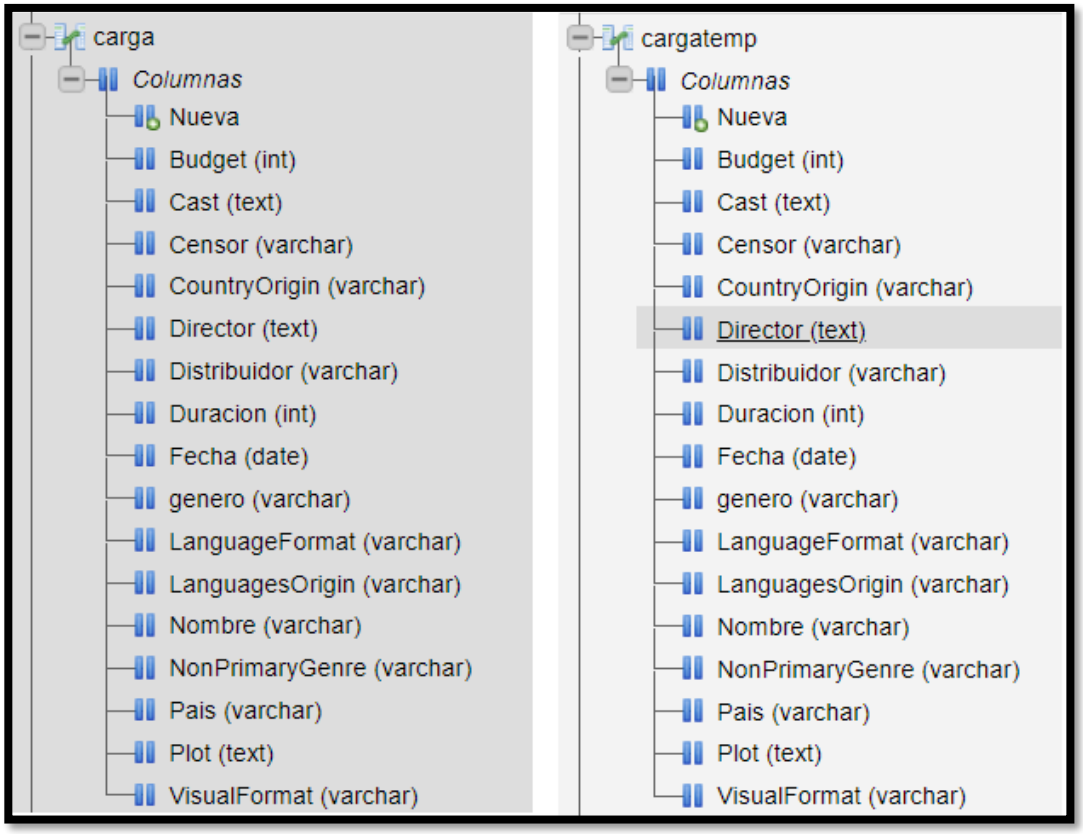
Transformación y validación de los datos.

La naturaleza altamente estructurada de la mayoría de los datos necesarios posibilita que el entrenamiento y prueba del modelo no requiera un proceso de transformación demasiado exigente en tiempo o complejidad de tratamiento. Cabe citar, sin embargo, como excepciones a este respecto, los conjuntos de datos a utilizar en PLN propios de los atributos de sinopsis, elenco, directores, etc. a los que accedemos a través de raspados en la red y otras técnicas automatizadas de recolección de datos.

- El trabajo de perfilado y limpieza es una tarea que lleva a cabo Comscore como proveedor de este tipo de información. En cuanto a la limpieza de los datos que obtenemos directamente de Google mediante un script, esta información se limpia directamente en el código Python del modelo.
- Se lleva a cabo la transformación y cálculo de nuevos datos en el dataset mediante código en Python.

Almacenamiento de los datos.

Se ha creado una base relacional con los datos de películas y cines ubicada en Azure para asistir a la interfaz del cliente en sus consultas y peticiones. El carácter estructurado de los datos ha facilitado el diseño y creación de esta base, que, además, simplifica la recogida de las consultas tanto a los usuarios de Filmstradamus 3000 desde el interfaz, como para el propio modelo para procesar dichas consultas.



Detalles de la carga.

Descubrimiento y modelado.

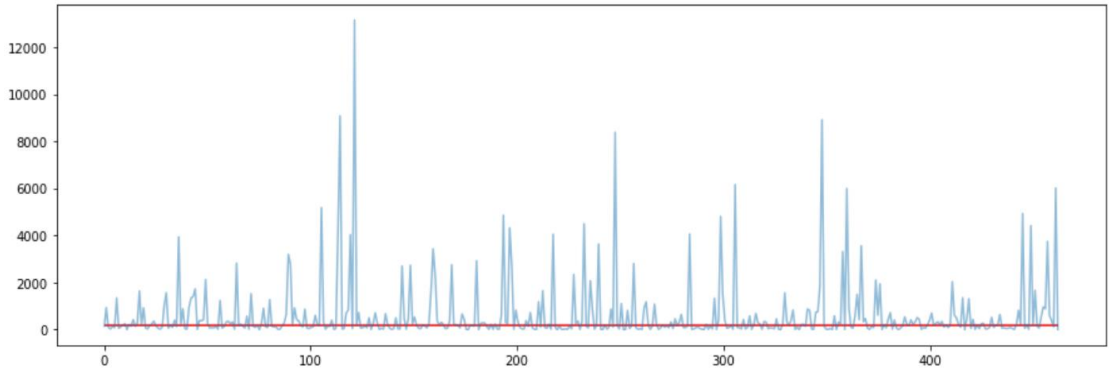
Una vez tenemos los datos de Comscore y Cinemark, y hemos completado la información necesaria mediante el módulo IMDBPy para recopilar datos referentes a elencos, sinopsis, y directores, integramos los datos para su tratamiento y transformación previa a su uso en el modelo. A continuación, ofrecemos una breve descripción del proceso.

- Solo se observan nulos en dos campos del conjunto inicial, ambos numéricos. Tras un estudio de las diferentes posibilidades, estos nulos se pueblan con el valor de la mediana del campo para todo el conjunto. También se encuentran nulos en el campo Non-primary genre, pero esto no será problemático puesto que vamos a combinarlos con Genre (que nunca es nulo) y aplicarles codificación (one-hot encoding) dada su condición de variables categóricas.

- Diferenciaremos entre las variables dependientes (aquellas solo conocidas una vez finaliza el ciclo de exhibición de la película, las relacionadas con asistencia y facturación) e independientes.

- Para la elección de las variables objetivo disponemos, dentro de nuestro grupo de variables dependientes, de varios campos relativos a facturaciones y asistencia (del primer fin de semana, de la primera semana, el total del ciclo de exhibición completo). Tras comprobar la alta correlación existente entre casi todas estas variables (exceptuando la de número de semanas en

cartel), se decide elegir una sola variable objetivo, que será la asistencia en espectadores del primer fin de semana de exhibición de la película. Se descarta la elección de la facturación en ese mismo período debido a las posibles fluctuaciones de los precios de las entradas, así como la variación de precios entre países, que no quedan registradas en el conjunto de datos.



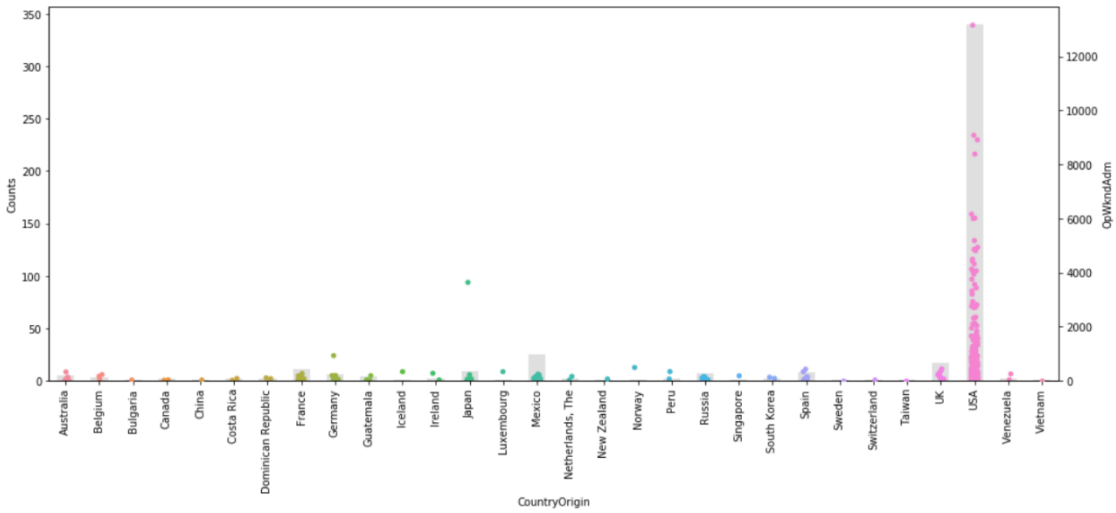
Valores de la variable objetivo (Op_Wknd_Adm) en el dataset (la línea roja señala la media de la variable).

- En las variables independientes se encuentran atributos tanto numéricos como categóricos.

- En las variables independientes de tipo numérico (Budget, Competition, Runtime) se explora la posibilidad de normalización (método maxmin), si bien se comprobará más adelante que dicho procedimiento no tendrá apenas incidencia en la calidad de la predicción.

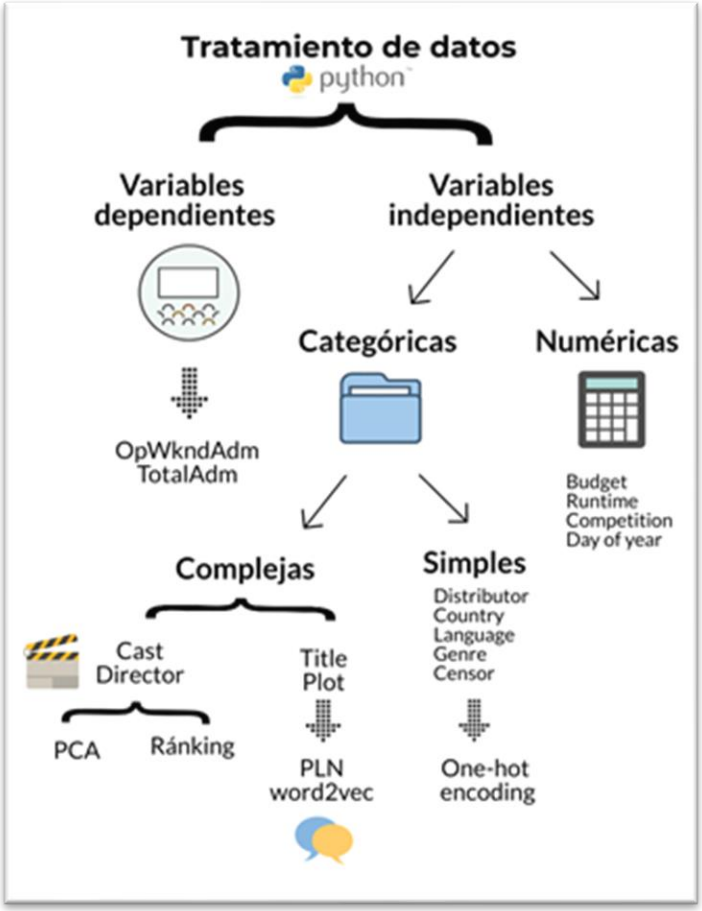
- Dentro de las variables categóricas vamos a explorar distintos enfoques, dependiendo de las características de las mismas:

<u>Variables *sencillas*:</u> (Distribuidora, País de origen, Idioma original, Géneros (primario + secundarios), etc.). Estas variables generalmente tienen 4-5 valores principales y el resto podría catalogarse dentro de una categoría Otros.



Cruzamos los valores de la variable objetivo (OpWkndAdm) con los distintos Países de origen (CountryOrigin), el área gris indica el número de películas de cada género.

Como ya hemos adelantado, en estas variables utilizaremos el método de one hot encoding, tras estudiar la representatividad de cada categoría y los umbrales necesarios para quedarnos con los más útiles para la predicción y reunir los menos en la categoría "Otros". No se prevén problemas de dimensionalidad dado el volumen relativamente reducido de datos, por lo que contamos con cierta flexibilidad en la elección de umbrales.



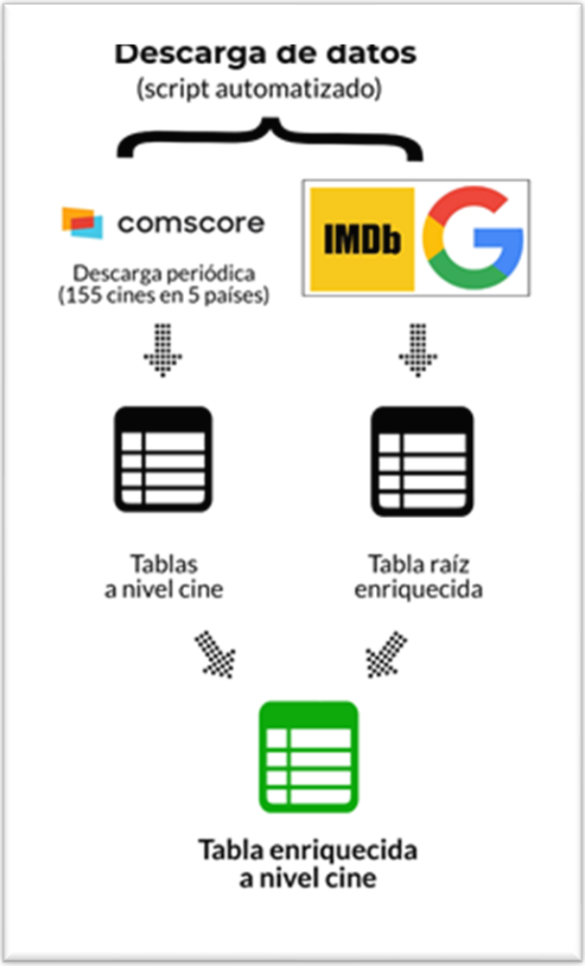
<u>Director y Elenco:</u> en este caso existen muchas categorías y, a priori, no parece sencillo delimitar qué directores o actores pueden tener importancia como característica predictiva. En este caso vamos a explotar dos enfoques muy distintos, pero que resultan complementarios en la implementación del modelo. Los detallamos a continuación.

- Ranking de Director/Elenco: se ordenarán actores y directores utilizando el valor medio de la variable objetivo de las películas en las que aparecen, situando a cada uno de ellos con un valor percentil (0-99) en un escalafón o ranking. A cada película se asignará un valor que consiste en la media de los pesos de cada uno de los actores del elenco (ídem para los directores). En el caso de las películas a predecir, la idea es realizar la media entre los actores que aparecen en nuestro escalafón, y si todos los actores son nuevos en el conjunto de datos, rellenar este campo con el percentil 50.

- Reducción de dimensionalidad para Director/Elenco: Debido a la alta dimensionalidad que la aplicación directa del método one hot encoding genera en ambos campos, dicha técnica se ha

complementado con el uso de PCA (Principal Component Analysis) para así poder reducir dicha dimensionalidad. Tras una serie de pruebas, llegamos a la conclusión de que el campo Elenco, por encima de 30 componentes principales (representa el 12.46% de información principal) la predicción deja de mejorar. Para el caso del campo Director, la predicción mejoró hasta los 20 componentes principales (el 9.08% de información principal).

Título y Plot: estos dos atributos son completamente distintos a los anteriores. Al tratarse de texto hemos decidido exprimir el potencial de las técnicas que ofrece PLN. En particular hemos explorado dos direcciones: la primera es la creación de un recomendador que relacione películas "similares" empleando una matriz de similitud tipo TfIdf generada por conteo de palabras. La segunda consiste, utilizando K-means, un algoritmo de clasificación no supervisada (clusterización) que agrupa objetos en k grupos basándose en sus características, agrupar las películas en clusters o segmentos en base a la similaridad argumental, que evaluaremos a través de algoritmos como Word2Vec o Doc2Vec, modelos de red neuronal que aprenden asociaciones de palabras dentro de un corpus de texto.



Parece que el empleo de Word2Vec mejora los resultados del modelo, al complementar la información ya explotada con un tamiz fino que parece lograr capturar cierta similitud entre películas (no siempre con rendimientos de taquilla equiparables).

- Una vez realizado este proceso, comenzamos a crear los diferentes conjuntos de entrenamiento, con diferentes combinaciones de variables predictivas (y sus umbrales de codificación en aquellas en que proceda) para evaluar la bondad de los diferentes modelos que se puedan considerar para realizar las predicciones. Este proceso nos ha permitido evaluar distintos modelos y a su vez encontrar qué características predictivas o atributos tienen importancia en la predicción.

- El siguiente paso consiste en unir los datos registrados en el nivel industria con los recogidos en el nivel sala. Obtenemos pues los datos por sala, que difieren de los de la industria en aspectos como nomenclatura de los atributos y otros, pero que son corregidos y unificados para poder unir ambos conjuntos.

- Los datos de cada sala de exhibición contienen un identificador de sala. Los datos generales de la industria contienen el nombre en USA de la película, y su nivel de granularidad es por país en el que se exhibe (una misma película cuenta con tantos registros como países de la región CAM en que se exhibe). A través de una función se crea un dataframe que combina los datos de facturación

del cine correspondiente con los datos asociados a las películas que ha proyectado, vinculando identificador y campos del cine y dataframe de las tablas raíz, a través de la tabla de equivalencias. A continuación, se exporta el DF a la Base de Datos de Azure con los siguientes campos [cine_name, cine_ID,cine_country, number_movies].

Una vez vinculados los registros, y tras llevar a cabo el proceso de limpieza, tratamiento de nulos si los hubiera resultado de la unión, etc., ya tenemos las tablas propias de cada sala con que realizar las predicciones a este nivel de granularidad, que es el buscado, pues se necesitan predicciones de cada película específicas para cada sala.

Pero antes de empezar a probar los modelos, debemos tener en cuenta una serie de consideraciones, principalmente, el volumen relativamente reducido de datos con los que contamos. En este contexto es importante considerar el riesgo de sobreajuste en que se puede incurrir. Algunas medidas tomadas para paliar este riesgo son:

- La utilización de modelos sencillos, o en caso de usar modelos más agresivos, manejar sus parámetros para mitigar el riesgo de sobreajuste.
- Tratamiento y detección de los valores atípicos; esta tarea aumenta en dificultad en conjuntos de datos tan dispersos como el que nos ocupa.
- Posiblemente prescindir de aquellos atributos que no aporten capacidad predictiva, haciendo un estudio de importancia.

Finalmente decidimos realizar validación cruzada en los datos a nivel cine, entrenando un modelo por cada uno de los cines de la región que tienen un número suficiente de registros (hemos decidido que el umbral se fije en 500 registros de películas, dado que por debajo de dicho nivel, el dataset se antoja demasiado pequeño para crear un modelo predictivo fiable). La exploración en esta fase nos ha hecho decidirnos por un modelo Random Forest y, tras un exhaustivo “grid search” hemos escogido los parámetros del mismo.

Propiedades del modelo

En este punto, hacemos un breve receso en la descripción técnica del modelo para describir el carácter y las propiedades del mismo en su dimensión de negocio y de aportación a la toma de decisiones.

- **Función descriptiva del modelo.**

• Si bien la empresa ya analizaba las variables que pueden afectar al desempeño de una película, este análisis se realizaba sin intervención de técnicas de aprendizaje automático. Al introducir ahora dichas técnicas en el proceso de toma de decisión y en el día a día, se potencia y facilita el análisis de las variables y la elaboración de indicadores y obtención de estadísticos más complejos y específicos.
• La aplicación de visualización desarrollada (inicialmente se ha optado por Tableau; en el siguiente apartado de este documento se incluye una amplia descripción) también permitirá a los miembros

del equipo poder entender y consultar esta información de forma mucho más inteligible y accesible. Creemos que esta aplicación de visualización servirá como la herramienta analítica ideal para enriquecer el conocimiento y la experiencia de negocio con que ya cuenta el equipo de Cinemark.

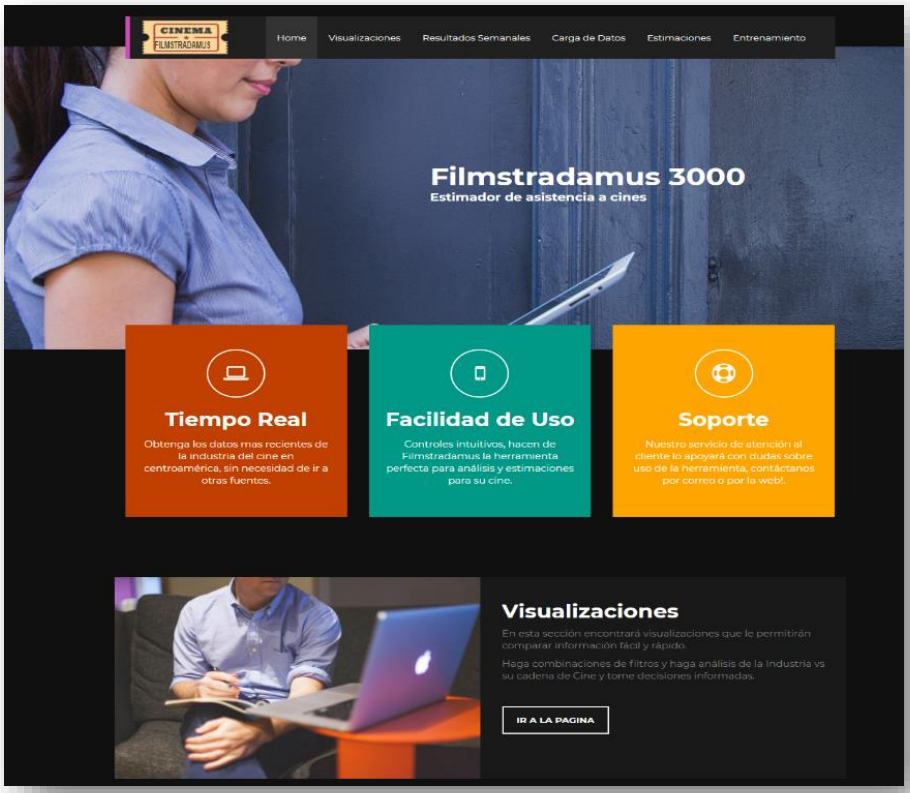
- **Función predictiva del modelo.**

- Su característica central y su propósito es la capacidad de mejorar la predicción de desempeño de las películas, y proveer de la información necesaria para explicar estas predicciones y sus grados y naturaleza de error.
- La propia naturaleza de los datos y por tanto de la variable a predecir complica la predicción. La similitud entre dos películas es difícilmente objetivable de por sí, pero lo es más si ha de medirse respecto a la variable objetivo, la asistencia de público. Dos películas muy similares en argumento, género cinematográfico, idioma, o productora (por citar algunos de los atributos estudiados) pueden diferir enormemente en cuanto a recaudación y asistencia. La complejidad de estas relaciones requiere un modelo automatizado de aprendizaje.
- La capacidad predictiva de estas variables en su estado original es baja. Muchas de ellas son variables categóricas que han requerido de codificación, y tanto para estas una vez codificadas como para el resto de variables independientes numéricas, se han desarrollado procesos de normalización y clustering o segmentación en aras de la afinación del modelo.
- El modelo también se diseña para explicar las diferencias de desempeño de una misma película según país o cine. La conjugación de datos de cine con datos globales de la industria hace posible que el desarrollo del modelo sirva a estos dos objetivos y aumente la especificidad del análisis y por tanto de las decisiones basadas en él.
- Esta especificidad sirve asimismo para facilitar la gestión de personal, suministros, etc. de las salas, la optimización de estos recursos depende en gran medida de tener una expectativa ajustada de la asistencia de público.

- **Función prescriptiva.**

La Programación de las películas, que por razones de tiempo, no se desarrolló en este primera fase del proyecto, pero forma parte de la solución que se busca conseguir toma como input el análisis descriptivo y predictivo, para optimizar los recursos de las salas de cine de Cinemark y aumentar la eficiencia Operativa.

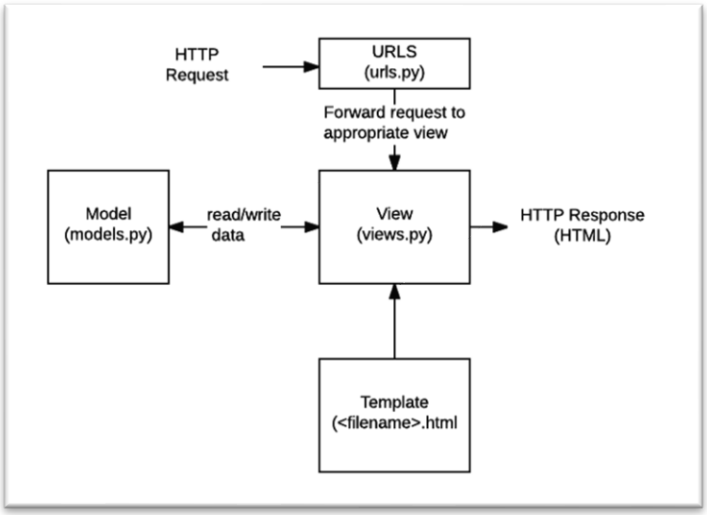
Funcionamiento del modelo.



Pantalla principal interfaz de usuario.

Retomamos la descripción del proceso de modelado con esta sección. En ella, se explica de forma más concreta desde el punto de vista técnico el proceso de funcionamiento de la herramienta. La interfaz de usuario está implementada en el framework web Django, el cual nos permite en el lado del cliente presentar html simple pero que haga llamadas a las funciones Python alojadas en el servidor.

La estructura de la implementación Django tiene la siguiente forma:



En la siguiente imagen podemos ver las diferentes urls y sus correspondientes views.

```
urlpatterns = [
    path('admin/', admin.site.urls),
    url(r'^$', views.hi),
    url(r'^home$', views.hi, name='home'),
    url(r'^resultados$', views.resultados, name='resultados'),
    url(r'^visualizaciones$', views.visualizaciones, name='visualizaciones'),

    url(r'^carga', views.carga, name='carga'),
    url(r'^carga$', views.carga, name='carga'),

    url(r'^estimaciones', views.estimaciones, name='estimaciones'),
    url(r'^estimaciones$', views.estimaciones, name='estimaciones'),
    url(r'^tabladump1', views.tabladump1, name='tabladump1'),
    url(r'^tabladump2', views.tabladump2, name='tabladump2'),
    url(r'^tabladump3', views.tabladump3, name='tabladump3'),
    url(r'^entrenamiento$', views.entrenamiento, name='entrenamiento'),
    url(r'^entrenamiento', views.entrenamiento_init, name='entrenamiento_init'),
    url(r'^visual1', views.visual1, name='visual1'),
    url(r'^buscar', views.buscar, name='buscar'),
    url(r'^busqueda', views.busqueda, name='busqueda'),
    url(r'^external', views.external),
    url(r'^modelo', views.modelo),
    url(r'^insertar', views.insertar),
    url(r'^archivotexto', views.archivotexto)
]
```

En cuanto a las funciones se han creado tres bloques de código principales, o rutinas, que conforman la secuencia o pipeline. El código está estructurado en funciones para ser invocadas desde la consulta del usuario. Estas rutinas pueden invocarse de forma independiente para su uso y revisión de los desarrolladores, pero están pensadas para ejecutarse en orden, como ocurrirá una vez en producción. Las rutinas son las siguientes:

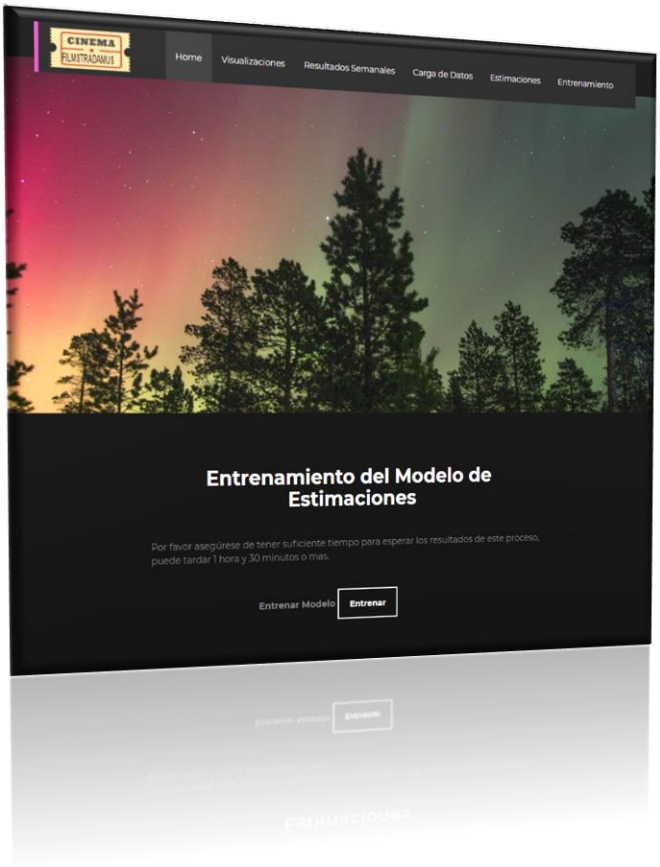
1. Rutinas.merge_all.

- Esta función toma la tabla de datos agregados, a partir del conjunto raíz, y la tabla con información general de los cines (data_key), y con los datos de ambos conjuntos y las tablas que se descargan automáticamente de Comscore, va creando los dataframes con la información a nivel. Más adelante se crearán los dataframes combinados de cine y agregados, y con ellos se entrenará el modelo.
- Desde data_key se llama en bucle a la información por cines, ordenando la llamada por países.
- En el conjunto raíz se llevará a cabo el siguiente procesamiento:
- Se crea el ranking de directores y elencos.
- Se limpian los campos de Título y de Argumento, que se utilizan para el análisis de lenguaje natural. Este paso se realiza en este punto y no en los archivos específicos de cada cine por motivos de rapidez y economía computacional. En función de los resultados de este análisis se crean los clusters de películas en función de la similaridad encontrada.
- Mediante una función, (merge_cine) completamos la unión de los archivos, combinando los dos niveles distintos de granularidad que aparecen en los datos originales antes de su tratamiento.
- En la última etapa de esta rutina, se exportan los archivos combinados Merge_ID para cada cine a su localización en la base. También se genera un archivo extra Cines en la misma

localización, que recoge un listado de los cines con la información ['Cine_ID','Cine_Name', 'Country', 'Number_movies'] para cada uno de ellos.

2. Rutinas.train_encode_all.

- Esta función llama a todos los archivos merge_cine con un número de registros mayor de 500. Para la codificación se elige un umbral de representatividad del 2%, por debajo del cual, el registro pertenece a la categoría conjunta alternativa "Other". Basándonos en las pruebas agregadas antes mencionadas, optamos por no normalizar las variables numéricas, tras comprobar que la normalización en cualquiera de sus formas no aporta ninguna mejora a la predicción.
- Se entrenan los modelos mediante validación cruzada empleando Random Forest, con los parámetros que se han obtenido del grid search en la fase exploratoria. Se guardarán para análisis futuros el RMSE de cada modelo a nivel cine, así como los joblib con los modelos, para generar las predicciones futuras.
- Se generan predicciones en entrenamiento y prueba, que se añaden a la tabla IDMerge. Se exportan los resultados a la base en IDPred.

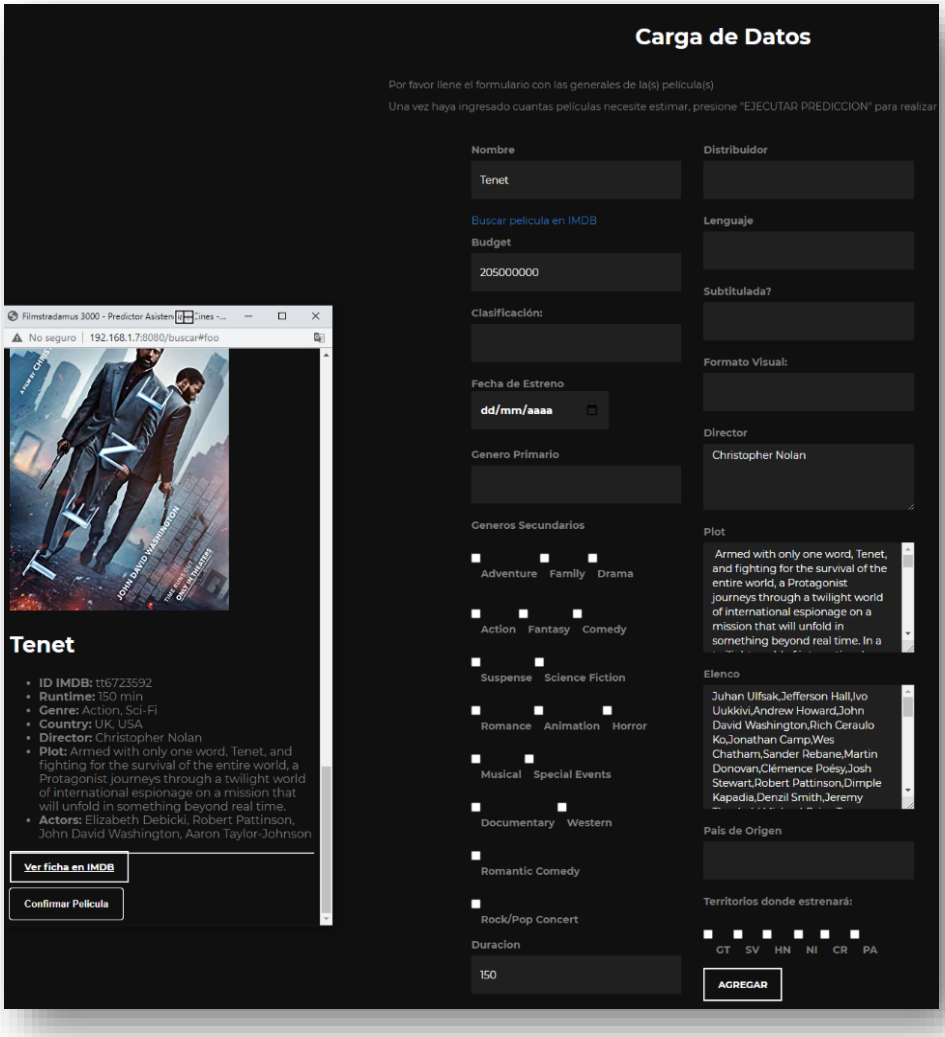


Pestaña de entrenamiento en la interfaz de usuario.

3. Rutinas.predicción

- Toma las consultas de los usuarios en la interfaz y crea un archivo NuevasPeliculas que contiene una serie de campos en común con campos que IDMerge (prescinde de las variables dependientes, de la objetivo, de la fecha de estreno y del número de películas en competencia a la fecha de estreno.

- Genera una predicción para cada cine en los países en los que se va a estrenar dicha película. Para poder realizar dicha predicción primero hay que calcular los campos faltantes y codificar la tabla cargada utilizando el mismo protocolo de codificación que se utilizó anteriormente



Interfaz para la carga de datos manual que conecta con IMDBPy

- Se carga el archivo IDModelRF.joblib, en el cual se genera la predicción. Se crea un archivo con la predicción para cada país, y uno agregado.

```
def archivotexto(request):
    if os.path.isfile('Data\\ArchivoNuevasPeliculas.csv'):
        os.remove('Data\\ArchivoNuevasPeliculas.csv')
    resultado = ObtenerArchivos.aCargarFinal()
    ObtenerArchivos.archivotexto(resultado)

    Rutinas.prediccion()
```

Captura de cómo se invocan las funciones descritas desde la interfaz

Algoritmo.

Como se ha dicho anteriormente, el volumen de datos disponible limitaba las posibilidades de elección de modelo para este proyecto. Finalmente hemos optado por un regresor por random forest, si bien en el proceso de pruebas hemos trabajado con otros métodos como xgboost, árboles de decisión y otros.

Debido a las características de nuestro modelo, la naturaleza de los datos, y el objetivo de negocio, la herramienta debe ser capaz de predecir con mayor precisión el comportamiento de los grandes estrenos, que técnicamente pueden ser considerados valores atípicos. Por este motivo, la métrica de referencia durante la construcción y el desarrollo del modelo has sido el error cuadrático medio.

Visualización.

La representación de la información y los mandos de control que permitirán hacer seguimiento a los resultados del modelo predictivo se articula a través de una solución web que integra todos los módulos, incluyendo el de visualización. Internamente los dashboard se han desarrollado usando la herramienta de Tableau Online que permite generar un link por cada dashboard creado.

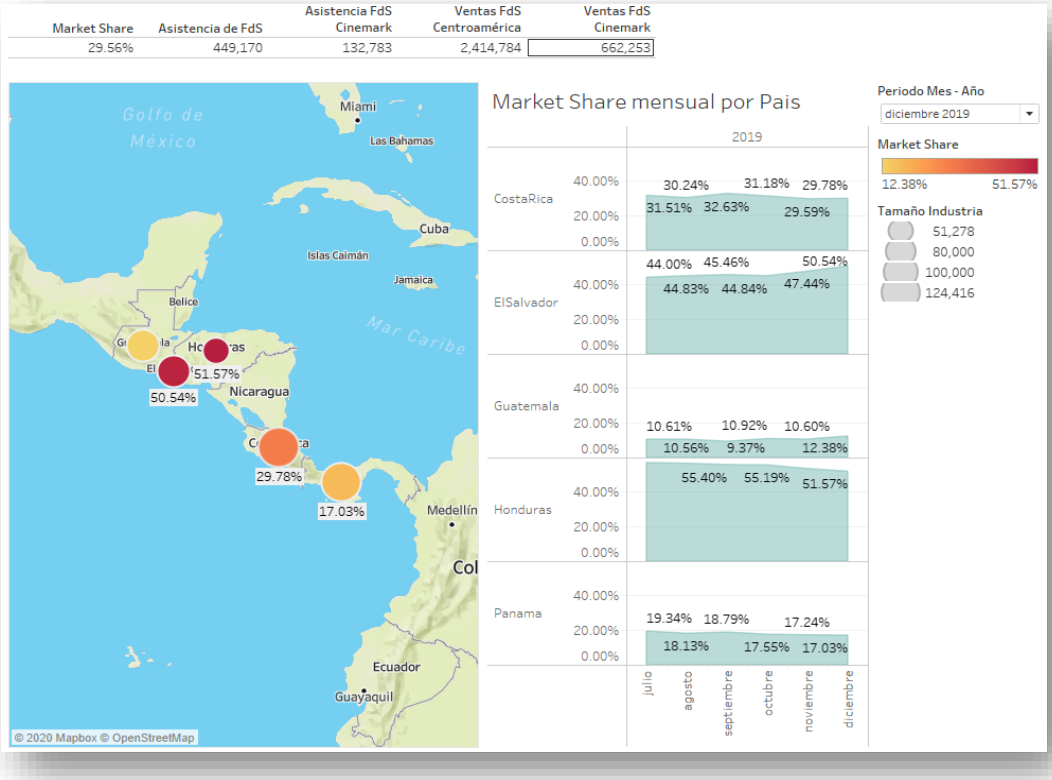
Una de las razones de escoger Tableau como herramienta de visualización es por su enfoque de autoservicio aplicado al análisis y toma de decisiones, gracias a la interactividad y dinámica que ofrece en sus cuadros de mando.

El objetivo que buscamos con la visualización es mostrar los datos de una forma muy visual, que permita encontrar los insight que se necesitan para tomar las decisiones de una manera más sencilla.

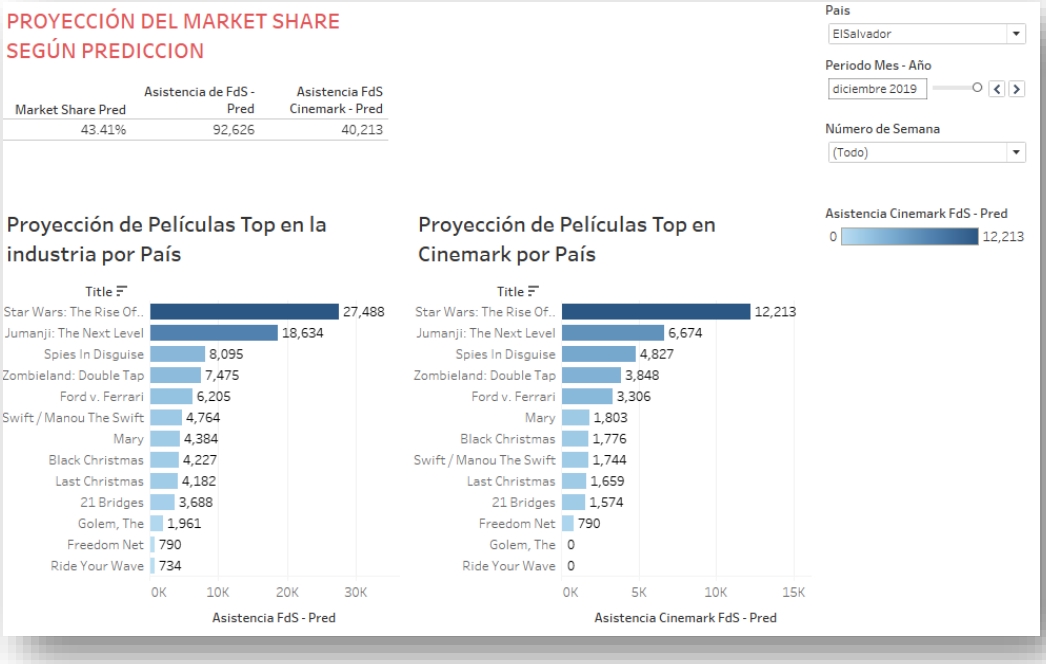
Tableau obtendrá toda la información que necesita desde la base de datos que se tiene en Azure, tanto como la información de las películas, la asistencia y ventas de todos los cines de la región como también la información de las predicciones.

Los mandos de Control que se van a disponer como parte de la solución son de 3 tipos:

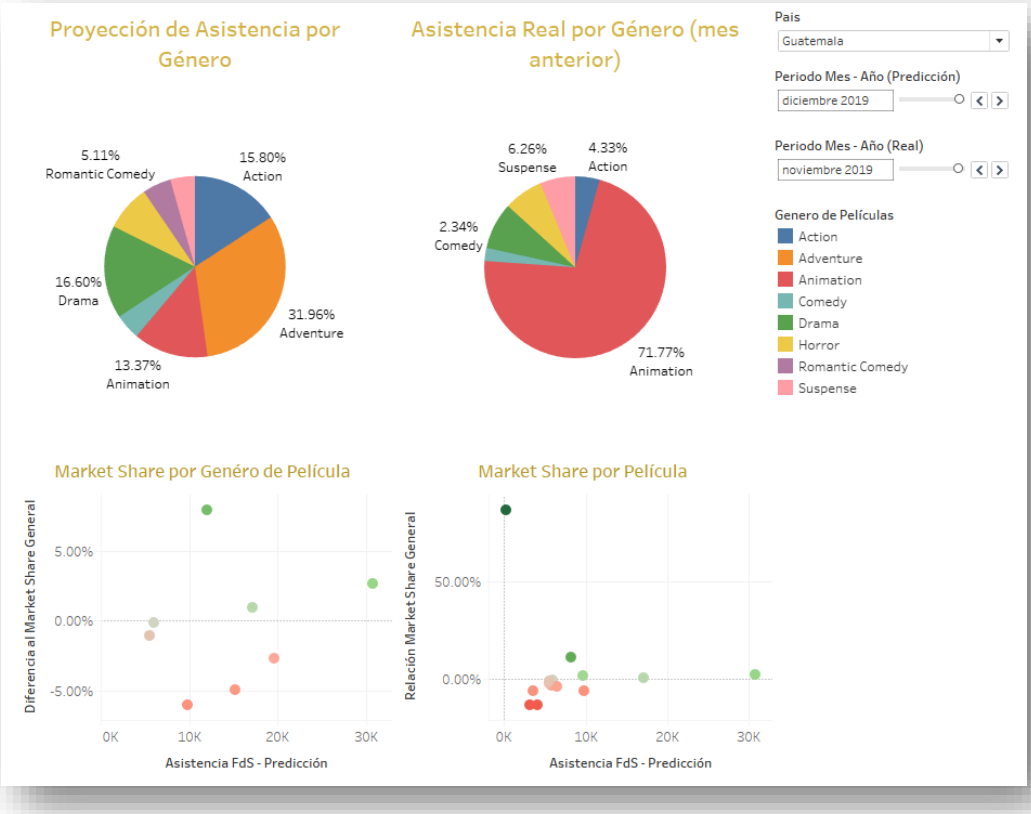
1- **Información General de la Industria.**
   Se complementa la información calculando el Market Share por país, mostrando el tamaño de la industria por país respecto a la cantidad de asistencia, la evolución del Market Share en el tiempo.



2- **Información de las Predicciones de las películas próximas a estrenarse.**
   Mostrando información de las películas Top a nivel País considerando a todos los cines de la industria o por las salas de Cinemark.



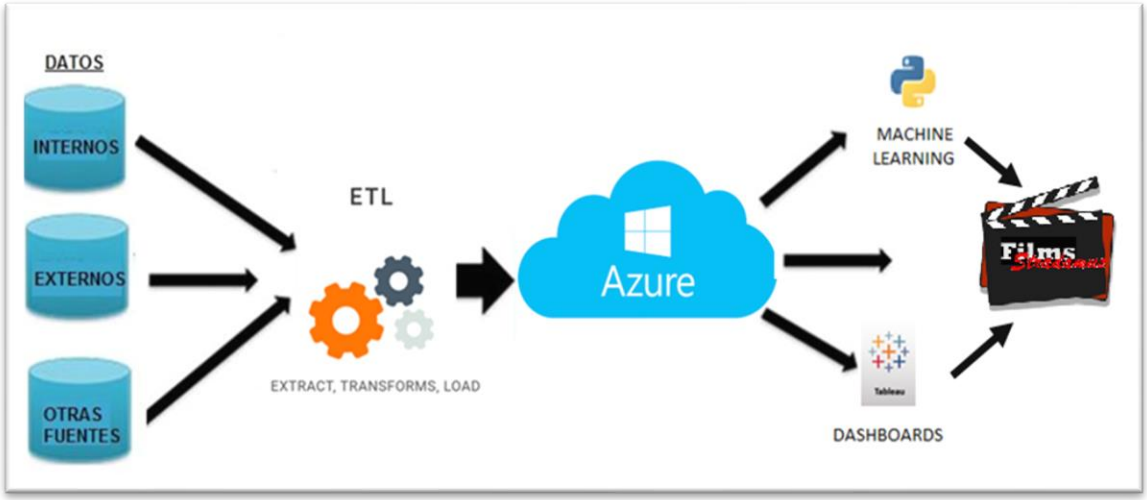
También podemos ver la cuota de mercado en base a la predicción de la cartelera a estrenar por género de películas y por películas.



3- **Información del modelo Predictivo, indicadores sobre la exactitud y precisión del modelo predictivo y de las predicciones hechas.**

En esta sección de las visualizaciones tenemos pensado por un lado guardar distintos valores relativos al modelo (RMSE, valor absoluto entre predicciones y datos reales, etc.) para evaluar el rendimiento del modelo y poder así implementar cambios si fuera necesario. Como estamos entrenando el modelo a nivel cine, se puede también obtener una comparativa de los distintos valores entre cines, o segmentando las predicciones por distintas variables: Género, Censor, Distribuidora, etc.

Solución tecnológica: arquitectura técnica.



Esquema de la solución

Análisis de recursos: talento humano y recursos físicos.

Estructura organizativa.

El proyecto requerirá el trabajo de 5 personas, de las que 2 serán de un perfil Analista/Desarrollador y otras 3 serán Consultor/Analista.

Denominación del recurso humano y tareas asignadas.

Ana Zumalacárregui Pérez (Analista, Desarrollador)

- Diseño y desarrollo del modelo predictivo.
- Limpieza de datos, ingeniería de atributos y búsqueda de características predictivas).
- Implementación del modelo).
- Desarrollo del modelo de PLN para aplicar a los títulos a la sinopsis de las películas.
- PCA reducción dimensional del one-hot-encoding.
- Diseño e implementación del pipeline con creación de funciones Python para la integración en el interfaz de usuario.
- Revisión y mejoras del modelo.

Dhani Geordie Montoya Rojas (Consultor, Analista)

- Búsqueda de características predictivas.
- PCA reducción dimensional del one-hot-encoding.
- Desarrollo de la visualización de los diferentes Dashboards del modelo a través de la herramienta de visualización Tableau.
- Conexión de Tableau con Google Sheets.

José Tellechea Mora (Consultor, Analista)

- Estudio y análisis de los datos de los diferentes modelos a aplicar en el proyecto.
- Estudio de parámetros del modelo y optimización del mismo.
- Búsqueda de características predictivas.
- Revisión y mejoras del modelo.
- Testeo del modelo final.
- Documentación del modelo.

Mario Alcides Bolaños Amaya (Analista, Desarrollador)

- Persona encargada de la comunicación entre Cinemark y el equipo de trabajo.
- Entrevista con responsable en Cinemark.
- Creación de la base de datos relacional que aglutina los diferentes datos necesarios para el modelo.
- Creación de un ranking para las variables director y actor.
- Desarrollo de la interfaz web para la carga de nuevos datos, ejecutar el modelo predictivo y mostrar el resultado del mismo (210 horas).
- Conexión de la interfaz web con Tableau Public para la visualización de los diferentes Dashboards.

Pablo Fernández Rodríguez (Consultor, Analista)

- Análisis preliminar de datos.
- Captura, tratamiento y preprocesamiento de los datos necesarios para ejecutar el modelo.
- Reuniones de seguimiento y coordinación con Cinemark.
- Testeo del modelo final.
- Documentación del modelo.

El costo de participación.

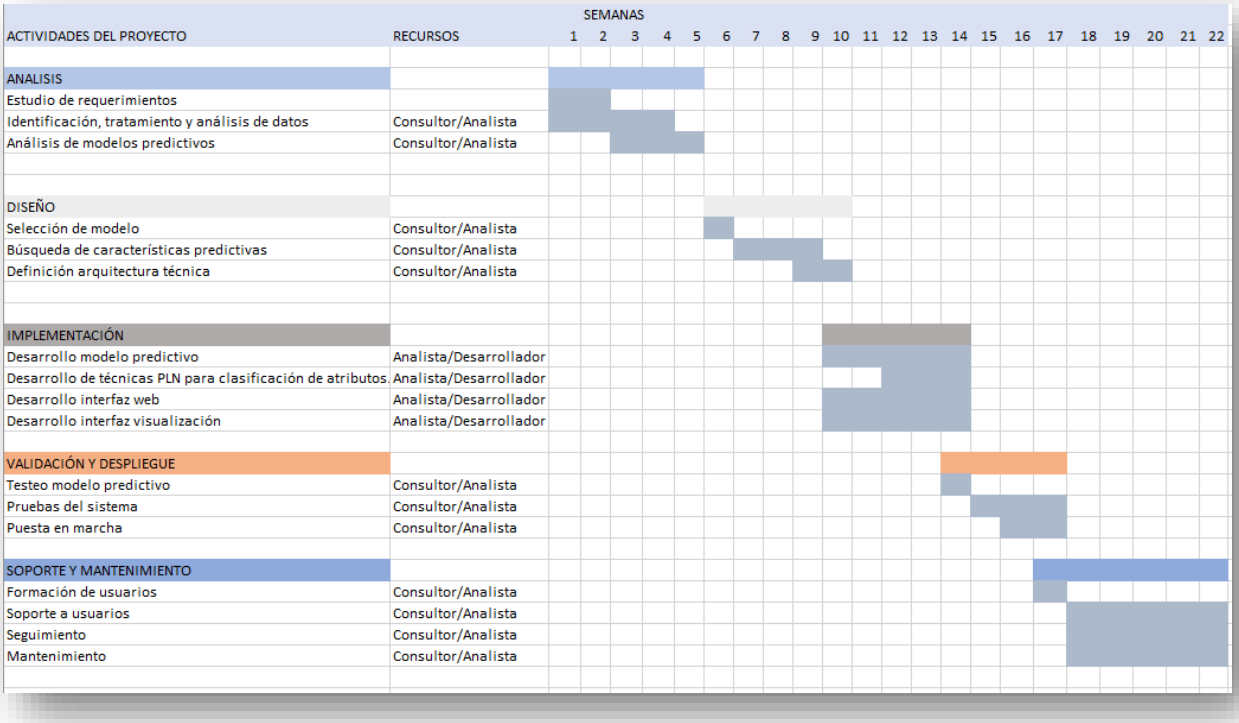
Infraestructura física (recursos físicos).

- **Ordenadores.**

Suministros y servicios externos.

- Servicio de Comscore.
- Servicio cloud Azure.
- Servicio Tableau Server.

Cronograma y reparto.



Rentabilidad proyecto.

En este apartado se va a detallar el análisis de viabilidad desde un punto de vista financiero. Para ello hemos realizado un estudio de las diferentes fuentes de beneficios y gastos con el fin de estudiar distintos indicadores de viabilidad como el VAN y la TIR. Se ha de tener en cuenta que la viabilidad del proyecto se estudia desde el punto de vista de Cinemark CAm, es decir, si para dicha compañía es rentable invertir en la solución que proponemos.

Es importante trasladar a Cinemark un aspecto fundamental de este proyecto y que pertenece al ámbito estrictamente financiero y de economía de la empresa, a pesar del carácter tecnológico del mismo. En primer lugar, una de las ventajas de los modelos de aprendizaje automático, aplicados al sector real, es que su implementación es poco costosa para los estándares de esta industria.

Asimismo, nos encontramos con otra ventaja añadida: la escalabilidad del proyecto. Cinemark es una empresa muy grande y asentada, y la solución que vamos a crear es aplicable a todos los ámbitos geográficos en los que opera la compañía. Si se quisiera extrapolar esta misma solución a otros mercados de Cinemark, el costo de traslado del modelo sería muy bajo.

A continuación, pasamos a enumerar algunos de los principales beneficios que generaría el proyecto, gracias a una mejora en la estimación de la asistencia (Fase I) y la consecuente programación óptima (Fase II) que llevaría a un aumento de cuota de mercado en la región.

Beneficios tangibles.

1. Generación de ingresos:
   - Aumento de la cuota de mercado, nuestra estimación es un incremento de un 1.5%
   - Aumento de la venta en restauración, derivado del incremento en asistencia previsto.
   - Mejora de la satisfacción de los clientes.

2. Reducción de costes:
   - Optimización de la gestión de compras perecederas y asignación de personal en los cines, debido a una mejor previsión de la asistencia.
   - Optimización del gasto energético.
   - Optimización de las acciones de marketing.

Beneficios intangibles.

- Mejora de la imagen de marca.
- Mejora del acceso a los datos a través de consultas, análisis o informes.
- Información más actualizada.
- Mayor aprovechamiento de los servicios de información contratados por Cinemark (Comscore).
- Incorporación y explotación de otras fuentes de datos públicas (IMDB) a través de procesos automatizados.

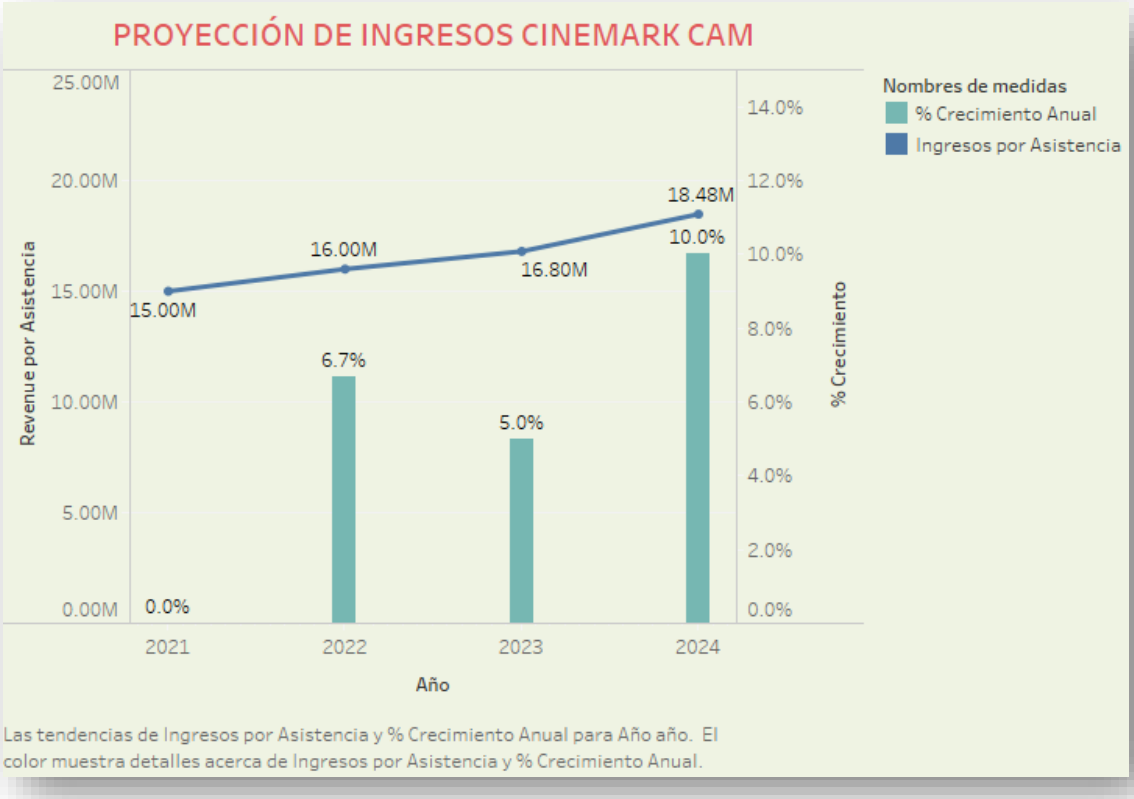
Beneficios estratégicos.

- Mejora del proceso de toma de decisiones, que se realizará de forma más rápida, informada y basada en datos.

Escenario económico.

A continuación, vamos a detallar las hipótesis o premisas de las que partimos para realizar nuestro análisis económico.

La situación de pandemia global en la que nos encontramos permite muy pocas certezas en cuanto a cómo se comportará la industria en los años venideros. Nos hemos basado en las propias previsiones que maneja Cinemark CAM. La compañía considera que el año próximo (2021) la facturación ascenderá de $15M, ligeramente por debajo del 2019, debido a que, a pesar de la situación de inestabilidad económica en la región, se han acumulado un gran número de estrenos que hace pensar que se tratará de un buen año para la industria en cuanto se reabran las salas. Hay que señalar que esta previsión asume como escenario la posibilidad de ocupación total de las salas de cine a partir de marzo de 2021. En este escenario, la compañía asume una cierta vuelta a la normalidad de cara al 2022 en adelante, años en los que Cinemark CAM prevé facturaciones similares a las previas a la pandemia.



La inversión inicial consistirá únicamente en la consultoría y desarrollo de la herramienta, con un costo estimado de US$135.000 con un gasto anual posterior entre US$30.000 y US$10.000 de mantenimiento de la herramienta. Además, se prevé un gasto único en compra de equipos de US$6.000. Se ha estimado la necesidad de 5 personas dedicadas al proyecto durante la fase de desarrollo (6 meses, a $25,96 la hora) y una de ellas durante la fase de implementación y mantenimiento posterior.

El primer año, 2020, contamos con que la herramienta no genere ningún beneficio por dos razones: en primer lugar las salas en la región han permanecido cerradas durante la mayor parte del desarrollo de la herramienta debido a la pandemia. En segundo lugar, incluso con la apertura de las salas los datos para explotar el modelo no se corresponden aún con los históricos que manejamos por la situación extraordinaria que vivimos.

Se asume que la herramienta no generará ingresos hasta el segundo trimestre de 2021, cuando se alcance cierta normalidad en la cartelera y contemos con un histórico de datos recientes para poder entrenar el modelo con mayor precisión.

ANÁLISIS DE RENTABILIDAD DEL PROYECTO					
	AÑO 0	AÑO 1	AÑO 2	AÑO 3	AÑO 4
INVERSIÓN					
Consultoría y desarrollo Filmstradamus	-$135,000	-$30,000	-$20,000	-$10,000	-$10,000
Equipos	-$6,000				
TOTAL INVERSIÓN	-$141,000	-$30,000	-$20,000	-$10,000	-$10,000
	AÑO 0	AÑO 1	AÑO 2	AÑO 3	AÑO 4
INGRESOS/BENEFICIOS					
Aumento de la cuota de mercado	$0	$22,500	$48,000	$77,400	$85,914
Aumento ventas en restauración	$0	$22,770	$48,272	$77,839	$86,402
Optimización de compras perecederas	$0	$22,500	$38,160	$51,278	$56,918
Optimización de la gestión de personal de los cines	$0	$22,800	$34,200	$45,600	$45,600
Optimización del gasto energético	$0	$5,400	$10,800	$13,500	$13,500
Optimización acciones marketing	$0	$28,175	$42,263	$56,350	$56,350
TOTAL INGRESOS/BENEFICIOS	$0	$124,145	$221,695	$321,967	$344,684
GASTOS	AÑO 0	AÑO 1	AÑO 2	AÑO 3	AÑO 4
Base de datos SQL en MySQL. Licencia libre en Azure	$1,500	$1,500	$1,500	$1,500	$1,500
Coste interfaz Godaddy	$36	$36	$36	$36	$36
Dominio	$12	$12	$18	$18	$18
TOTAL GASTOS	$1,548	$1,548	$1,554	$1,554	$1,554
FLUJO DE CAJA OPERATIVO	-$1,548	$122,597	$220,141	$320,413	$343,130
FLUJOS DE CAJA	-$142,548	$92,597	$200,141	$310,413	$333,130
VAN: 1%	$759,153				
VAN: 5%	$637,511				
VAN: 10%	$516,170				
VAN: 6.02% WACC de Cinemark a 10/2020	$610,263				
TIR:	108%				

Análisis de ingresos.

Aumento de la cuota de mercado.

Hemos estimado que con las predicciones de un modelo ajustado podemos adelantarnos y ajustar las reprogramaciones que venían produciéndose los sábados. Esto supondría un aumento en cuota de mercado entre 1% y 2% de los días previos a la reprogramación (jueves y viernes) y que suponen en torno al 30% de la facturación semanal.
Se considera que durante el Año 1 (2021) obtendremos un incremento del 0.5% en la cuota de mercado sobre las previsiones de Cinemark CAM ($15M para 2021 en la región) debido a la necesidad de mejorar y adaptar el modelo a la nueva situación. Durante el siguiente año, dicho aumento se verá incrementado hasta un 1% de crecimiento de cuota de mercado y finalmente esperamos en los Años 3 y 4 obtener un incremento del 1.5%. hemos calculado que con el aumento de cuota de mercado generado por la implementación del modelo podemos esperar un incremento del consumo de aproximadamente la mitad del aumento de la cuota de mercado (0.25% Año 1 - 0.75% Años 3 y 4).

Aumento en las ventas de restauración (Alimentos y Bebidas).

Sobre una facturación en este concepto de $2.3M/mes en la región hemos calculado que con el aumento de cuota de mercado generado por la implementación del modelo podemos esperar un incremento del consumo de aproximadamente la mitad del aumento de la cuota de mercado (0.25% Año 1 – 0.75% Años 3 y 4).

Optimización de compras perecederas.

Cinemark CAM gasta aproximadamente $500.000 al mes en este tipo de productos. Estimamos que con una mejor estimación de la asistencia, este gasto se puede optimizar dando una ganancia de entre un 0.37% y un 0.75% (dependiendo de la precisión de la herramienta).

Optimización de personal en los cines.

Estimamos con datos de la empresa que se tienen en media 25 empleados a tiempo parcial por cine, a $400/mes, y se puede reducir este gasto un 1%-2% con una mejor estimación de la asistencia que se traduce en una gestión de personal más eficiente.

Optimización del gasto energético.

Cinemark CAM se gasta un promedio de $4.5M al año en energía. La optimización que se puede hacer la podemos considerar según el departamento de operaciones únicamente en funciones que dejen de darse por no tener suficientes asistentes como para cubrir los costos de energía que se da por utilizar la sala. Esto en promedio en los años de 2016 a 2019 representó el 0.20% de todas las funciones de cada año. Hemos supuesto que con la mejora en las estimaciones que vamos a obtener, podemos doblar este porcentaje.

Optimización de acciones de marketing.

El gasto anterior en publicidad era de $45.000 mensuales en la región. Si se usa ese presupuesto de manera estratégica en productos promocionales de las películas que impactarán a mayor cantidad de gente podríamos obtener un retorno de inversión que calcularemos con la siguiente información:

- Costo del producto promocional $1.10/ud.
- Precio de venta: $2.25/ud.
- Tiraje: 100,000 unidades por película (5 películas al año).

Normalmente se vende el 70% del tiraje, la meta sería lograr vender un 20% más, es decir, un 84% del tiraje.

Análisis de gastos.

Los gastos reales del proyecto por parte de Cinemark CAM, una vez completado el desarrollo y la implementación, se reducen a:

- Licencia para una base de datos SQL en MySQL. Licencia (libre) en Azure.
- Hosting.

- Dominio.

No incluimos en gastos la licencia de Tableau, que requiere la implementación de la herramienta, porque la empresa ya cuenta con la misma y por tanto no sería un gasto imputable al proyecto.

Indicadores de valoración de la inversión. VAN/TIR.

Para el proyecto Filmstradamus 3000, la inversión inicial es de $141.000. Los flujos de caja netos de los cinco años que hemos considerado son los siguientes:

Año 0	Año 1	Año 2	Año 3	Año 4
-$142.548	$92.597	$200.141	$310.413	$333.130

En este escenario, el payback o plazo de retorno de la inversión, se da alrededor del segundo trimestre de 2022 (Año 2).

Presentamos diferentes tipos de descuento en el cálculo del VAN para exponer cómo afecta a este indicador dicha elección.

FLUJO DE CAJA OPERATIVO	-$1,548
FLUJOS DE CAJA	-$142,548
VAN: 1%	$759,153
VAN: 5%	$637,511
VAN: 10%	$516,170
VAN: 6.02% WACC de Cinemark a 10/2020	$610,263
TIR:	108%

SIn embargo, para el VAN estimado para el proyecto, hemos utilizado como tipo de descuento el coste medio de capital (WACC) de Cinemark, 6.02% a julio de 2020, resultando en un valor actual de $610.263.

Se ha considerado la posibilidad de añadir una prima a este tipo, práctica habitual cuando se quiere reflejar algún riesgo adicional respecto del genérico de la empresa. Esta posible prima se la atribuimos a tratar de implantarse esta herramienta inicialmente en una región económica como Centro América, de mayor riesgo percibido que la de Estados Unidos, en la que Cinemark tiene su sede, y país en el que cotizan sus acciones. En cualquier caso, al no haber debatido este punto con

la compañía, y por evitar una valoración de dicha prima acaso inexacta, se asume la aplicación del tipo genérico antes mencionado, que refleja exactamente el coste medio de capital.

En este escenario, la TIR del proyecto queda estimada en un 108%, lo que refleja las posibilidades de una herramienta de coste muy contenido, aplicable a los aspectos del negocio más determinantes en la cuenta de resultados. Es digno de mención el hecho de la alta transferibilidad tanto del predictor de asistencias como de la herramienta de programación a otras regiones de negocio de Cinemark a un coste de implementación muy bajo.

Bibliografía y recursos.

- Aurélien Géron, O´Reilly (2017). Hands-On Machine Learning with Scikit-Learn and TensorFlow, Concepts, Tools, and Techniques to Build Intelligent Systems.
- Demand-Driven Scheduling of Movies in a Multiplex (Eliashberg, Hegie, Ho, Huisman, Miller, Swami, Weinberg, Wierenga) https://repub.eur.nl/pub/10069
- Comportamiento del consumidor de cine en salas: factores motivacionales y tipología del consumidor (Gil Martín, Mª Montserrat 2018) https://eprints.ucm.es/46080/
- Ainslie, A., Dreze, X., & Zufryden, F. (2005). Modeling movie lifecycles and market share.Marketing Science, 24(3), 508-517.
- Andrews, R. L., Currim, I. S., Leeflang, P., & Lim, J. (2008). Estimating the Scan∗Pro model of store sales: HB, FM or just OLS? International Journal of Research in Marketing, 25, 22-33.
- Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P., & Vance, P. H. (1998).
- Danaher, P. J., & Mawhinney, D. (2001). Optimizing television program schedules using choice modeling. Journal of Marketing Research, 38(3), 298-312.
- Eliashberg, J., Jonker, J. J., Sawhney, M. S., & Wierenga, B. (2000). MOVIEMOD: An implementable decision support system for prerelease market evaluation of motion pictures. Marketing Science, 19(3), 226-243.
- Eliashberg, J., Swami, S., Weinberg, C. B., & Wierenga, B. (2001). Implementing and evaluating SilverScreener: A marketing management support system for movie Exhibitors. Interfaces, 31, S108-S127 (May–June).
- Horen, Jeffrey H. (1980). Scheduling of network television programs. Management
- Science, 26, 354-370 (April).
- Reddy, S. K., Aronson, J. E., & Stam, A. (1998). SPOT: Scheduling programs optimally for television. Management Science, 44(1), 83-102.
- Sawhney, M. S., & Eliashberg, J. (1996). A parsimonious model for forecasting gross box office revenues of motion pictures. Marketing Science, 15(2), 113-131.
- Swami, S., Eliashberg, J., & Weinberg, C. B. (1999). SilverScreener: A modeling approach to movie screens management. Marketing Science, 18(3), 352-372.
- Wierenga, B., & Oude Ophuis, P. A. M. (1997). Marketing decision support systems:
- Adoption, use and satisfaction. International Journal of Research in Marketing, 14(3),275-290.

Anexos.

Anexo I

Proyecto Web
Código Tratamiento de Datos

Anexo II

Código Machine Learning: Rutinas y funciones

Anexo III

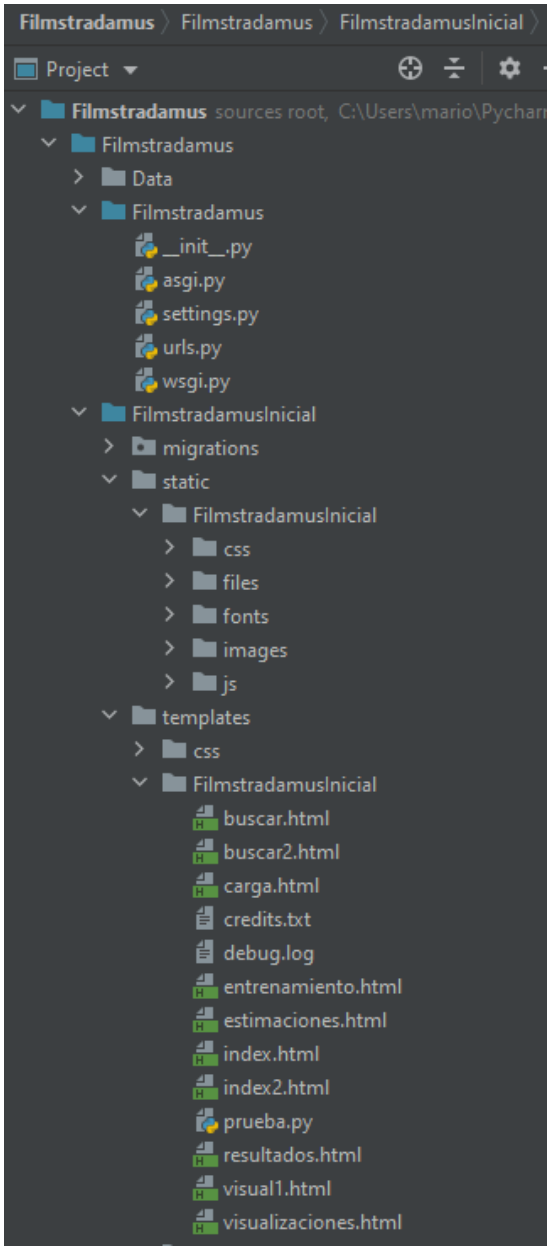
Instrumentos de visualización y cuadros de mando

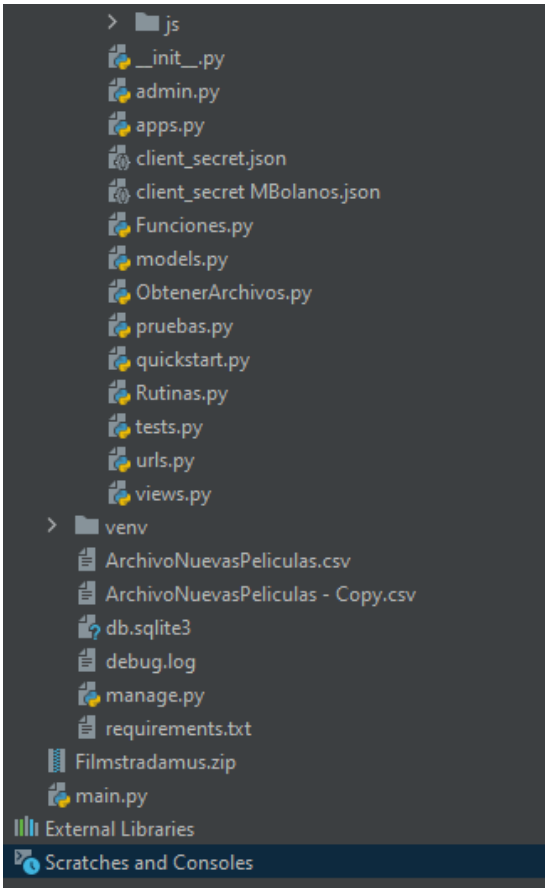
ANEXO I

Proyecto Web

Código Tratamiento de Datos

Proyecto Web Django





Código Tratamiento de Datos

ObtenerArchivos.py

En este archivo:

Funciones prox_jueves, sacarURL, semanas y obtenerarchivos se utilizan en conjunto para, a partir de una fecha que se pasa como parámetro a la función obtenerarchivos obtener la URL que se abrirá en el navegador para descargar los reportes que correspondan a la semana que se busque desde la interfaz web, tanto para Cinemark como para la industria, de la misma manera la función prox_jueves calcula las fechas que corresponde al jueves anterior, al mismo jueves del año anterior y el jueves anterior del año anterior. Con estos datos y con las URLs para descargar esos ocho archivos la función semanas puede proceder a bajarlos al folder destinado para eso y renombrarlos para que obtenerarchivos haga el proceso ETL de datos.

Funciones DBquery, DBquerySelect, DBqueryDelete, aCargar y aCargarFinal se utilizar para realizar consultas a la base de datos,

La función IMDB recibe como parámetro el Id de IMDB de una película y devuelve toda la información que tenga registrada la película en la base de datos de este recurso web, se usa para rellenar los campos de la carga de las películas a estimar.

La función conexiongoogle hace la escritura en los archivos de Google sheets que usa la interfaz para conectar con las visualizaciones de Tableau online y tener actualización en tiempo real de las mismas.

```
import time
import pandas as pd
import selenium
import sys
import os
import datetime as dt
import time as t
import mysql.connector
import imdb
import re
from . import Rutinas
import warnings

import gspread
from oauth2client.service_account import ServiceAccountCredentials
from gspread_dataframe import get_as_dataframe, set_with_dataframe
import gspread_dataframe as gd
from mysql.connector import Error
from datetime import date, datetime, timedelta
from dateutil.relativedelta import relativedelta
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from pandas import DataFrame
from pathlib import Path


path = Path(__file__)

warnings.filterwarnings("ignore")
```

```
connection = mysql.connector.connect(host='localhost', database='filmstradamus',
user='root', password='')

levels_up = 2
directorio = path.parents[levels_up - 1]
csp = str(Path(directorio))

driverchrome = csp + "\\Data\\chromedriver.exe"
fecha = sys.argv[1]

def prox_jueves(d, weekday):
    days_ahead = weekday - d.weekday()
    if days_ahead <= 0:  # Target day already happened this week
        days_ahead += 7
    return d + dt.timedelta(days_ahead)


def sacarURL(fecha, chunk, chunk2):
    anio = str(fecha.year)
    fechamt = fecha.month
    fechady = fecha.day

    if fechamt < 10:
        mes = '0' + str(fechamt)
    else:
        mes = str(fechamt)

    if fechady < 10:
        dia = '0' + str(fechady)
    else:
        dia = str(fechady)

    fechaenviada = anio + '-' + mes + '-' + dia
    URL = chunk + fechaenviada + chunk2
    return URL


def semanas(x, URL, compania):
    import time
    chrome_options = webdriver.ChromeOptions()
    prefs = {'download.default_directory': csp + "\\Data\\reportes"}
    chrome_options.add_experimental_option('prefs', prefs)
    driver = webdriver.Chrome(driverchrome, options=chrome_options)
    # driver.get('https://www.iboe.com/')

    optionsIND = {0: 'TYTW',
                  1: 'TYLW',
                  2: 'LYTW',
                  3: 'LYLW'
                  }

    optionsCNMK = {0: 'CNMK TYTW',
                   1: 'CNMK TYLW',
                   2: 'CNMK LYTW',
                   3: 'CNMK LYLW'
                   }

    if compania == 0:
        quesemana = optionsIND[x]
    else:
        quesemana = optionsCNMK[x]

    driver.get(URL)

    driver.find_element_by_name('login_id').send_keys('UsrsCNMK15')
    driver.find_element_by_name('password').send_keys('Passwd12')
    driver.find_element_by_id('beta-supported').click()

    # time.sleep(10)
    element = WebDriverWait(driver, 100).until(
        EC.element_to_be_clickable(
            (By.XPATH, "//span[@class='pull-right btn-toolbar']//span[@class='fa fa-lg fa-
file-excel-o']")))
    driver.find_element_by_xpath(
```

```
        "//span[@class='pull-right btn-toolbar']//span[@class='fa fa-lg fa-file-excel-
o']").click()

    while not os.path.exists(csp+'\\Data\\reportes\\Top Territories_Top Titles.xls'):
        time.sleep(1)

    if os.path.isfile(csp+'\\Data\\reportes\\Top Territories_Top Titles.xls'):
        os.rename(csp+'\\Data\\reportes\\Top Territories_Top Titles.xls',
                  csp+'\\Data\\reportes\\' + quesemana + '.xls')
    driver.quit()


def DBquery(consulta, valores, tipo):
    with connection.cursor() as cursor:
        cursor.execute(consulta, valores)
        if tipo == 1:
            rows = cursor.fetchall()
            return rows
        else:
            connection.commit()


def DBquerySelect(consulta):
    with connection.cursor() as cursor:
        cursor.execute(consulta)
        rows = cursor.fetchall()
    return rows


def DBqueryDelete(consulta):
    with connection.cursor() as cursor:
        cursor.execute(consulta)
        connection.commit()


def aCargar():
    rows = DBquerySelect("SELECT * FROM cargatemp")
    return rows


def aCargarFinal():
    rows = DBquerySelect("SELECT * FROM carga")
    return rows


def insertar(pais, nombre, budget, distribuidor, release_date, genero, duracion,
             VisualFormat, NonPrimaryGenre, Censor, CountryOrigin, LanguagesOrigin,
             LanguageFormat, Director, Cast, Plot):
    datos = ({'Pais': pais,
              'Nombre': nombre,
              'Budget': budget,
              'Distribuidor': distribuidor,
              'Release_date': release_date,
              'Genero': genero,
              'Duracion': duracion,
              'VisualFormat': VisualFormat,
              'NonPrimaryGenre': NonPrimaryGenre,
              'Censor': Censor,
              'CountryOrigin': CountryOrigin,
              'LanguagesOrigin': LanguagesOrigin,
              'LanguageFormat': LanguageFormat,
              'Director': Director,
              'Cast': Cast,
              'Plot': Plot
              })

    columnas = ['pais', 'nombre', 'budget', 'distribuidor', 'fecha', 'genero', 'duracion',
                'VisualFormat', 'NonPrimaryGenre', 'Censor', 'CountryOrigin',
'LanguagesOrigin',
                'LanguageFormat', 'Director', 'Cast', 'Plot']

    valores = tuple(list(datos.values()))
    cols = ",".join([str(i) for i in columnas])
```

```
    DBquery("INSERT INTO cargatemp (" + cols + ") VALUES (" + "%s," * (len(datos) - 1) +
"%s);", tuple(datos.values()),
            0)
    return cols


def obtenerarchivos(fecha):
    format_string = "%Y-%m-%d"
    fechaactual = dt.datetime.strptime(fecha, format_string)
    fechaselect = (fecha,)
    DFrames = pd.DataFrame()
    rows = DBquery("Select * from resultados_historicos where fecha = %s", fechaselect, 1)

    if os.path.isfile(csp+'\\Data\\reportes\\TYLW.xls'):
        os.remove(csp+'\\Data\\reportes\\TYLW.xls')
        os.remove(csp+'\\Data\\reportes\\LYLW.xls')
        os.remove(csp+'\\Data\\reportes\\TYTW.xls')
        os.remove(csp+'\\Data\\reportes\\LYTW.xls')
        os.remove(csp+'\\Data\\reportes\\CNMK TYLW.xls')
        os.remove(csp+'\\Data\\reportes\\CNMK LYLW.xls')
        os.remove(csp+'\\Data\\reportes\\CNMK TYTW.xls')
        os.remove(csp+'\\Data\\reportes\\CNMK LYTW.xls')

    if len(rows) == 0:
        delta = relativedelta(years=0, months=0, days=-7, hours=0, minutes=0)
        delta1 = relativedelta(years=-1, months=0, days=0, hours=0, minutes=0)
        delta2 = relativedelta(years=-1, months=0, days=-7, hours=0, minutes=0)

        lastweek = fechaactual + delta
        lastyrweek = fechaactual + delta1
        lastyrltweek = fechaactual + delta2

        LY = prox_jueves(lastyrweek, 3)  # 0 = Monday, 1=Tuesday, 2=Wednesday...
        LYLW1 = prox_jueves(lastyrltweek, 3)  # 0 = Monday, 1=Tuesday, 2=Wednesday...

        fechas = [fechaactual, lastweek, LY, LYLW1]

        chunk0 =
"https://iboe.com/app.html#reports/film_grosses/top_territories_top_titles/country_id=24&di
stributor_id=ALL&day_range_rev="
        chunk1 =
"https://iboe.com/app.html#reports/film_grosses/top_territories_top_titles/country_id=24&di
stributor_id=ALL&coc_no=4481&day_range_rev="
        chunk2 =
"&filter_by_release_date_option=0&top_x_countries=all_countries&top_y_titles=all_titles&gro
ss_or_xtns=rev_and_xtns&english_or_local_title=aka&currency_type_no=1&revenue_day_option=we
ekend&pct_change_type=prev_week&country_of_origin=ALL&original_language=ALL&show_advanced_o
ptions=1"

        chunks = [chunk0, chunk1]

        z = 0
        for y in chunks:
            x = 0
            for i in fechas:
                envio = sacarURL(i, y, chunk2)
                semanas(x, envio, z)
                x += 1
            z += 1

        TYLW = pd.read_excel(csp+"\\Data\\reportes\\TYLW.xls", usecols="A,C,N,R,T,X",
skiprows=[0, 1])
        TYLW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm', 'WkndAdm']
        TYLW['fecha'] = lastweek.date()
        TYLW['industria'] = 1

        LYLW = pd.read_excel(csp+"\\Data\\reportes\\LYLW.xls", usecols="A,C,N,R,T,X",
skiprows=[0, 1])
        LYLW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm', 'WkndAdm']
        LYLW['fecha'] = LYLW1.date()
        LYLW['industria'] = 1

        TYTW = pd.read_excel(csp+"\\Data\\reportes\\TYTW.xls", usecols="A,C,N,R,T,X",
skiprows=[0, 1])
```

```
        TYTW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm', 'WkndAdm']
        TYTW['fecha'] = fechaactual.date()
        TYTW['industria'] = 1

        LYTW = pd.read_excel(csp+"\\Data\\reportes\\LYTW.xls", usecols="A,C,N,R,T,X",
skiprows=[0, 1])
        LYTW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm', 'WkndAdm']
        LYTW['fecha'] = LY.date()
        LYTW['industria'] = 1

        CNMK_TYLW = pd.read_excel(csp+"\\Data\\reportes\\CNMK TYLW.xls",
usecols="A,C,N,R,T,X",
                                  skiprows=[0, 1])
        CNMK_TYLW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm',
'WkndAdm']
        CNMK_TYLW['fecha'] = lastweek.date()
        CNMK_TYLW['industria'] = 0

        CNMK_LYLW = pd.read_excel(csp+"\\Data\\reportes\\CNMK LYLW.xls",
usecols="A,C,N,R,T,X",
                                  skiprows=[0, 1])
        CNMK_LYLW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm',
'WkndAdm']
        CNMK_LYLW['fecha'] = LYLW1.date()
        CNMK_LYLW['industria'] = 0

        CNMK_TYTW = pd.read_excel(csp+"\\Data\\reportes\\CNMK TYTW.xls",
usecols="A,C,N,R,T,X",
                                  skiprows=[0, 1])
        CNMK_TYTW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm',
'WkndAdm']
        CNMK_TYTW['fecha'] = fechaactual.date()
        CNMK_TYTW['industria'] = 0

        CNMK_LYTW = pd.read_excel(csp+"\\Data\\reportes\\CNMK LYTW.xls",
usecols="A,C,N,R,T,X",
                                  skiprows=[0, 1])
        CNMK_LYTW.columns = ['Pais', 'Pelicula', 'JuevesGBO', 'WkndGBO', 'JuevesAdm',
'WkndAdm']
        CNMK_LYTW['fecha'] = LY.date()
        CNMK_LYTW['industria'] = 0

        frames = [TYLW, LYLW, TYTW, LYTW, CNMK_TYLW, CNMK_LYLW, CNMK_TYTW, CNMK_LYTW]
        frametotal = pd.concat(frames)

        fechasdfs = []
        for fechadfs in fechas:
            fechadfs = fechadfs.date()
            rows = DBquery("Select * from resultados_historicos_peliculas where fecha =
%s", (fechadfs,), 1)
            if len(rows) == 0:
                fechasdfs.append(fechadfs)

        # frametotal['fecha'] = pd.to_datetime(frametotal['fecha'])
        todb = frametotal[frametotal.fecha.isin(fechasdfs)]
        cols = ",".join([str(i) for i in todb.columns.tolist()])

        for i, row in todb.iterrows():
            DBquery(
                "INSERT INTO resultados_historicos_peliculas (" + cols + ") VALUES (" +
"%s," * (len(row) - 1) + "%s);",
                tuple(row), 0)

        conexiongoogle("2daViz", todb, 'a')

        gpTYLW = TYLW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpTYLW["WkndGBO"] = round(gpTYLW["WkndGBO"], 2)
        gpLYLW = LYLW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpLYLW["WkndGBO"] = round(gpLYLW["WkndGBO"], 2)
        gpTYTW = TYTW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpTYTW["WkndGBO"] = round(gpTYTW["WkndGBO"], 2)
        gpLYTW = LYTW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpLYTW["WkndGBO"] = round(gpLYTW["WkndGBO"], 2)
```

```
        gpCNMK_TYLW = CNMK_TYLW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpCNMK_TYLW["WkndGBO"] = round(gpCNMK_TYLW["WkndGBO"], 2)
        gpCNMK_LYLW = CNMK_LYLW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpCNMK_LYLW["WkndGBO"] = round(gpCNMK_LYLW["WkndGBO"], 2)
        gpCNMK_TYTW = CNMK_TYTW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpCNMK_TYTW["WkndGBO"] = round(gpCNMK_TYTW["WkndGBO"], 2)
        gpCNMK_LYTW = CNMK_LYTW.groupby(["Pais"])[["WkndGBO", "WkndAdm"]].apply(sum)
        gpCNMK_LYTW["WkndGBO"] = round(gpCNMK_LYTW["WkndGBO"], 2)

        DFrames = [gpTYLW, gpLYLW, gpTYTW, gpLYTW, gpCNMK_TYLW, gpCNMK_LYLW, gpCNMK_TYTW,
gpCNMK_LYTW]

        from functools import reduce
        DFrames = reduce(lambda left, right: pd.merge(left, right, on=['Pais'],
                                                      how='outer'), DFrames)

        DFrames.columns = ['GBOTYLW', 'AdmTYLW', 'GBOLYLW', 'AdmLYLW', 'GBOTYTW',
'AdmTYTW', 'GBOLYTW', 'AdmLYTW',
                           'CNMK_GBOTYLW', 'CNMK_AdmTYLW', 'CNMK_GBOLYLW', 'CNMK_AdmLYLW',
'CNMK_GBOTYTW',
                           'CNMK_AdmTYTW', 'CNMK_GBOLYTW', 'CNMK_AdmLYTW']

        DFrames['fecha'] = fecha

        cols = ",".join([str(i) for i in DFrames.columns.tolist()])
        cols = cols + ",pais"

        for i, row in DFrames.iterrows():
            DBquery("INSERT INTO resultados_historicos (" + cols + ") VALUES (" + "%s," *
(len(row)) + "%s);",
                    tuple(row) + (i,), 0)

        DFramesSol = DBquery("Select * from resultados_historicos where fecha = %s",
fechaselect, 1)

        DFrameSol = DataFrame(DFramesSol, columns=[
            'pais', 'fecha', 'GBOTYLW', 'AdmTYLW', 'GBOLYLW', 'AdmLYLW', 'GBOTYTW',
'AdmTYTW', 'GBOLYTW', 'AdmLYTW',
            'CNMK_GBOTYLW', 'CNMK_AdmTYLW', 'CNMK_GBOLYLW', 'CNMK_AdmLYLW', 'CNMK_GBOTYTW',
            'CNMK_AdmTYTW', 'CNMK_GBOLYTW', 'CNMK_AdmLYTW'])

        conexiongoogle("2daVizHistorico", DFrameSol, 'a')

    else:

        DFrames = pd.DataFrame(rows, columns=['Pais', 'fecha', 'GBOTYLW', 'AdmTYLW',
'GBOLYLW', 'AdmLYLW', 'GBOTYTW',
                                              'AdmTYTW',
                                              'GBOLYTW', 'AdmLYTW', 'CNMK_GBOTYLW',
'CNMK_AdmTYLW', 'CNMK_GBOLYLW',
                                              'CNMK_AdmLYLW',
                                              'CNMK_GBOTYTW', 'CNMK_AdmTYTW',
'CNMK_GBOLYTW', 'CNMK_AdmLYTW'])

    DFrames.reset_index(inplace=True)

    ms = round(DFrames['CNMK_AdmTYTW'].sum() / DFrames['AdmTYTW'].sum() * 100, 2)
    gboms = round(DFrames['CNMK_GBOTYTW'].sum() / DFrames['GBOTYTW'].sum() * 100, 2)
    lyms = round(DFrames['CNMK_AdmLYTW'].sum() / DFrames['AdmLYTW'].sum() * 100, 2)
    lygboms = round(DFrames['CNMK_GBOLYTW'].sum() / DFrames['GBOLYTW'].sum() * 100, 2)

    resultado = DFrames[["Pais", "AdmTYTW", "CNMK_AdmTYTW"]].append(
        DFrames[["AdmTYTW", "CNMK_AdmTYTW"]].sum().rename('Total'))
    resultado['Share'] = round((resultado["CNMK_AdmTYTW"] / resultado["AdmTYTW"]) * 100, 2)

    return ms, lyms, gboms, lygboms, resultado, fecha


def archivotexto(resultado):
    df = pd.DataFrame(resultado,
                      columns=['Territory', 'Title', 'Budget', 'Dist',
'WeekofFirstEngagement', 'Genre', 'Runtime',
                               'VisualFormat', 'NonPrimaryGenre', 'Censor',
'CountryOrigin',
```

```
                                'LanguagesOrigin', 'LanguageFormat', 'Director', 'Cast',
'Plot'])

    df.to_csv(csp+'\\Data\\ArchivoNuevasPeliculas.csv',
              mode='a', index=False)

    DBqueryDelete("DELETE FROM carga")
    return resultado


def tabladumpfunc():
    DBqueryDelete("INSERT INTO carga Select * from cargatemp")
    DBqueryDelete("DELETE FROM cargatemp")
    return "done"


def tabladumpfunc2():
    DBqueryDelete("DELETE FROM cargatemp")
    return "done"


def tabladumpfunc3():
    DBqueryDelete("DELETE FROM carga")
    return "done"


def training():
    Rutinas.merge_all()
    Rutinas.train_encode_all()
    mensaje = "Entrenamiento Realizado"
    return mensaje


def IMDB(idmovie):
    moviesDB = imdb.IMDb()
    plot = ''
    synopsis = ''
    movie = moviesDB.get_movie(idmovie)

    title = movie['title']

    if 'directors' in movie.keys():
        director = movie['directors']
        directStr = ','.join(map(str, director))
    else:
        directStr = ''

    if 'cast' in movie.keys():
        casting = movie['cast']
        actores = ','.join(map(str, casting))
    else:
        actores = ''

    if 'plot' in movie.keys():
        for line in movie['plot']:
            line = line.split('::')[0]
            plot = plot + " " + line
    else:
        plot = ''

    if 'synopsis' in movie.keys():
        for line1 in movie['synopsis']:
            line1 = line1.split('::')[0]
            synopsis = synopsis + " " + line1
    else:
        synopsis = ''

    if 'runtimes' in movie.keys():
        runtime = movie['runtimes'][0]
    else:
        runtime = ''

    if 'box office' in movie.keys():
        if 'Budget' in movie['box office']:
```

```
            budget = movie['box office'].get('Budget')
            budget = ''.join(filter(str.isdigit, budget))
        else:
            budget = ''
    else:
        budget = ''

    sinopsis = plot + ' ' + synopsis

    return title, budget, runtime, directStr, sinopsis, actores


def conexiongoogle(archivo, df, mode='r'):
    directorio2 = csp + "\\FilmstradamusInicial\\client_secret.json"
    scope = ['https://www.googleapis.com/auth/spreadsheets',
"https://www.googleapis.com/auth/drive"]
    creds = ServiceAccountCredentials.from_json_keyfile_name(directorio2, scope)
    client = gspread.authorize(creds)

    sheet = client.open(archivo).sheet1

    if (mode == 'w'):
        sheet.clear()
        gd.set_with_dataframe(worksheet=sheet, dataframe=df, include_index=False,
include_column_header=True,
                              resize=True)
        return True
    elif (mode == 'a'):
        sheet.add_rows(df.shape[0])
        gd.set_with_dataframe(worksheet=sheet, dataframe=df, include_index=False,
include_column_header=False,
                              row=sheet.row_count + 1, resize=False)
        return True
    else:
        dfr = gd.get_as_dataframe(worksheet=sheet)
        return dfr
```

Views.py

En este archivo las funciones hi, resultados, visualizaciones, entrenamiento, entrenamiento_init, visual, buscar, modelo se usan para hacer llamadas a los elementos HTML de la aplicación.

La función estimaciones hace la llamada a los procedimientos que corren el modelo contra las películas que se han cargado previamente y escribe los resultados en la base de datos y los muestra en pantalla.

La función external muestra en la página html el contenido de las queries que obtienen información de cada país desde la tabla que almacena resultados históricos.

Las funciones tabledump1, tabledump2 y tabledump3 se usan para eliminar los contenidos de las tablas temporales donde se escribe la información de las películas a estimar.

```
from django.shortcuts import render
from pandas import DataFrame
import requests
```

```
import sys
import os
from subprocess import run, PIPE
import pandas as pd
import json

from . import Rutinas
from . import ObtenerArchivos

from django.http import HttpResponse
from django.template import loader
from django.shortcuts import redirect
from . import ObtenerArchivos
from django.http import HttpResponse
from django.template.response import TemplateResponse
from pathlib import Path

path = Path(__file__)

levels_up = 2
directorio = path.parents[levels_up - 1]
csp = str(Path(directorio))

def hi(request):
    return render(request, 'FilmstradamusInicial/index.html')

def resultados(request):
    return render(request, 'FilmstradamusInicial/resultados.html')

def visualizaciones(request):
    return render(request, 'FilmstradamusInicial/visualizaciones.html')

def carga(request):
    resultado = ObtenerArchivos.aCargar()
    df = DataFrame(resultado,columns=['Pais', 'Nombre', 'Budget', 'Distribuidor',
'FechaEstreno', 'Genero', 'Duracion',
                                      'VisualFormat', 'NonPrimaryGenre', 'Censor',
'CountryOrigin', 'LanguagesOrigin',
                                      'LanguageFormat', 'Director','Cast', 'Plot'])

    allData = []
    for i in range(df.shape[0]):
        temp = df.iloc[i]
        allData.append(dict(temp))

    cargatemp = {'data': allData}
    return render(request, 'FilmstradamusInicial/carga.html', cargatemp)

def estimaciones(request):
    resultado = ObtenerArchivos.aCargarFinal()
    df = DataFrame(resultado,columns=['Pais', 'Nombre', 'Budget', 'Distribuidor',
'FechaEstreno', 'Genero', 'Duracion',
                                      'VisualFormat', 'NonPrimaryGenre', 'Censor',
'CountryOrigin', 'LanguagesOrigin',
                                      'LanguageFormat', 'Director','Cast', 'Plot'])


    preds=ObtenerArchivos.conexiongoogle("Predicciones",df)
    allData0 = []
    for i in range(preds.shape[0]):
        temp = preds.iloc[i]
        allData0.append(dict(temp))

    allData = []
    for i in range(df.shape[0]):
        temp = df.iloc[i]
        allData.append(dict(temp))

    carga = {'datas': allData, 'historicos': allData0}
    return render(request, 'FilmstradamusInicial/estimaciones.html', carga)

def entrenamiento(request):
    return render(request, 'FilmstradamusInicial/entrenamiento.html')
```

```
def entrenamiento_init(request):
    mensaje = ObtenerArchivos.training()
    return render(request, 'FilmstradamusInicial/entrenamiento.html', {'mensaje': mensaje})

def visual1(request):
    return render(request, 'FilmstradamusInicial/visual1.html')

def buscar(request):
    return render(request, 'FilmstradamusInicial/buscar.html')

def modelo(request):
    return render(request, 'FilmstradamusInicial/visual1.html')

def external(request):
    inp = request.POST.get('param')
    out, out1, out2, out3, resultado, fecha = ObtenerArchivos.obtenerarchivos(inp)
    #print(result)
    df = DataFrame(resultado,columns=['Pais', 'AdmTYTW', 'CNMK_AdmTYTW', 'Share'])
    df = df.to_html
    fechas = (fecha,fecha)
    fechasgt = (fecha, 'GT', fecha, 'GT')
    fechassv = (fecha, 'SV', fecha, 'SV')
    fechashn = (fecha, 'HN', fecha, 'HN')
    fechasni = (fecha, 'NI', fecha, 'NI')
    fechascr = (fecha, 'CR', fecha, 'CR')
    fechaspa = (fecha, 'PA', fecha, 'PA')

    queryselect = "SELECT a.pelicula as 'Top 5', sum(a.wkndadm) - b.cinemark as 'Industry',
b.cinemark as 'Cinemark', round((b.cinemark/(sum(a.wkndadm) - b.cinemark))*100,2) as Share
FROM `resultados_historicos_peliculas` as a LEFT JOIN (select b.pelicula as movie,
sum(b.wkndadm) as cinemark from `resultados_historicos_peliculas` as b where b.fecha= %s
and b.industria = 0 GROUP by b.pelicula) as b ON a.pelicula = b.movie where a.fecha= %s
GROUP by a.pelicula order by 2 desc LIMIT 5"

    queryselectgt = "SELECT a.pelicula as 'Top 5', sum(a.wkndadm) - b.cinemark as
'Industry', b.cinemark as 'Cinemark', round((b.cinemark/(sum(a.wkndadm) -
b.cinemark))*100,2) as Share FROM `resultados_historicos_peliculas` as a LEFT JOIN (select
b.pelicula as movie, sum(b.wkndadm) as cinemark from `resultados_historicos_peliculas` as b
where b.fecha= %s and b.industria = 0 and b.pais = %s GROUP by b.pelicula) as b ON
a.pelicula = b.movie where a.fecha= %s and a.pais = %s GROUP by a.pelicula order by 2 desc
LIMIT 5"
    queryselectsv = "SELECT a.pelicula as 'Top 5', sum(a.wkndadm) - b.cinemark as
'Industry', b.cinemark as 'Cinemark', round((b.cinemark/(sum(a.wkndadm) -
b.cinemark))*100,2) as Share FROM `resultados_historicos_peliculas` as a LEFT JOIN (select
b.pelicula as movie, sum(b.wkndadm) as cinemark from `resultados_historicos_peliculas` as b
where b.fecha= %s and b.industria = 0 and b.pais = %s GROUP by b.pelicula) as b ON
a.pelicula = b.movie where a.fecha= %s and a.pais = %s GROUP by a.pelicula order by 2 desc
LIMIT 5"
    queryselecthn = "SELECT a.pelicula as 'Top 5', sum(a.wkndadm) - b.cinemark as
'Industry', b.cinemark as 'Cinemark', round((b.cinemark/(sum(a.wkndadm) -
b.cinemark))*100,2) as Share FROM `resultados_historicos_peliculas` as a LEFT JOIN (select
b.pelicula as movie, sum(b.wkndadm) as cinemark from `resultados_historicos_peliculas` as b
where b.fecha= %s and b.industria = 0 and b.pais = %s GROUP by b.pelicula) as b ON
a.pelicula = b.movie where a.fecha= %s and a.pais = %s GROUP by a.pelicula order by 2 desc
LIMIT 5"
    queryselectni = "SELECT a.pelicula as 'Top 5', sum(a.wkndadm) - b.cinemark as
'Industry', b.cinemark as 'Cinemark', round((b.cinemark/(sum(a.wkndadm) -
b.cinemark))*100,2) as Share FROM `resultados_historicos_peliculas` as a LEFT JOIN (select
b.pelicula as movie, sum(b.wkndadm) as cinemark from `resultados_historicos_peliculas` as b
where b.fecha= %s and b.industria = 0 and b.pais = %s GROUP by b.pelicula) as b ON
a.pelicula = b.movie where a.fecha= %s and a.pais = %s GROUP by a.pelicula order by 2 desc
LIMIT 5"
    queryselectcr = "SELECT a.pelicula as 'Top 5', sum(a.wkndadm) - b.cinemark as
'Industry', b.cinemark as 'Cinemark', round((b.cinemark/(sum(a.wkndadm) -
b.cinemark))*100,2) as Share FROM `resultados_historicos_peliculas` as a LEFT JOIN (select
b.pelicula as movie, sum(b.wkndadm) as cinemark from `resultados_historicos_peliculas` as b
where b.fecha= %s and b.industria = 0 and b.pais = %s GROUP by b.pelicula) as b ON
a.pelicula = b.movie where a.fecha= %s and a.pais = %s GROUP by a.pelicula order by 2 desc
LIMIT 5"
    queryselectpa = "SELECT a.pelicula as 'Top 5', sum(a.wkndadm) - b.cinemark as
'Industry', b.cinemark as 'Cinemark', round((b.cinemark/(sum(a.wkndadm) -
b.cinemark))*100,2) as Share FROM `resultados_historicos_peliculas` as a LEFT JOIN (select
b.pelicula as movie, sum(b.wkndadm) as cinemark from `resultados_historicos_peliculas` as b
where b.fecha= %s and b.industria = 0 and b.pais = %s GROUP by b.pelicula) as b ON
```

```
a.pelicula = b.movie where a.fecha= %s and a.pais = %s GROUP by a.pelicula order by 2 desc
LIMIT 5"

    top5 = ObtenerArchivos.DBquery(queryselect, tuple(fechas), 1)
    top5df = DataFrame(top5,columns=['Peliculas', 'Industria', 'Cinemark', 'Share'])
    top5df = top5df.to_html

    top5gt = ObtenerArchivos.DBquery(queryselectgt, tuple(fechasgt), 1)
    top5dfgt = DataFrame(top5gt,columns=['Peliculas', 'Industria', 'Cinemark', 'Share'])
    top5dfgt = top5dfgt.to_html

    top5sv = ObtenerArchivos.DBquery(queryselectsv, tuple(fechassv), 1)
    top5dfsv = DataFrame(top5sv,columns=['Peliculas', 'Industria', 'Cinemark', 'Share'])
    top5dfsv = top5dfsv.to_html

    top5hn = ObtenerArchivos.DBquery(queryselecthn, tuple(fechashn), 1)
    top5dfhn = DataFrame(top5hn,columns=['Peliculas', 'Industria', 'Cinemark', 'Share'])
    top5dfhn = top5dfhn.to_html

    top5ni = ObtenerArchivos.DBquery(queryselectni, tuple(fechasni), 1)
    top5dfni = DataFrame(top5ni,columns=['Peliculas', 'Industria', 'Cinemark', 'Share'])
    top5dfni = top5dfni.to_html

    top5cr = ObtenerArchivos.DBquery(queryselectcr, tuple(fechascr), 1)
    top5dfcr = DataFrame(top5cr,columns=['Peliculas', 'Industria', 'Cinemark', 'Share'])
    top5dfcr = top5dfcr.to_html

    top5pa = ObtenerArchivos.DBquery(queryselectpa, tuple(fechaspa), 1)
    top5dfpa = DataFrame(top5pa,columns=['Peliculas', 'Industria', 'Cinemark', 'Share'])
    top5dfpa = top5dfpa.to_html

    return render(request,'FilmstradamusInicial/resultados.html',
                  {'ms': out,'lyms': out1,'gboms': out2,'lygboms': out3,
                   'resultado': df, 'fecha': fecha, 'top5': top5df,
                   'top5gt': top5dfgt,'top5sv': top5dfsv,'top5hn': top5dfhn,
                   'top5ni': top5dfni,'top5cr': top5dfcr,'top5pa': top5dfpa})


def tabladump1(request):
    tabla = ObtenerArchivos.tabladumpfunc()
    response = redirect('/estimaciones/')
    return response


def tabladump2(request):
    print("tabladump2")
    ObtenerArchivos.tabladumpfunc2()
    return render(request, 'FilmstradamusInicial/carga.html')


def tabladump3(request):
    tabla = ObtenerArchivos.tabladumpfunc3()
    response = redirect('/carga/')
    return response

def insertar(request):
    pais = request.POST.getlist('extras')
    nombre = request.POST.get('movie_name')
    budget = request.POST.get('budget')
    distribuidor = request.POST.get('distribuidor')
    release_date = request.POST.get('fecha')
    genero = request.POST.get('genero')
    duracion = request.POST.get('duracion')
    VisualFormat = request.POST.get('3D')
    NonPrimaryGenre = request.POST.getlist('gensec')
    Censor = request.POST.get('censor')
    CountryOrigin = request.POST.get('pais')
    LanguagesOrigin = request.POST.get('idiomas')
    LanguageFormat = request.POST.get('sub')
    Director = request.POST.get('director')
    Cast = request.POST.get('cast')
    Plot = request.POST.get('plot')

    generosec = ""
```

```
    for gs in NonPrimaryGenre:
        generosec = gs + "," + generosec

    for p in pais:
        ObtenerArchivos.insertar(p, nombre, budget, distribuidor, release_date, genero,
duracion,
                                 VisualFormat, generosec, Censor, CountryOrigin,
LanguagesOrigin,
                                 LanguageFormat, Director,Cast, Plot)

    resultado = ObtenerArchivos.aCargar()
    df = DataFrame(resultado,columns=['Pais', 'Nombre', 'Budget', 'Distribuidor',
'FechaEstreno', 'Genero', 'Duracion',
                                      'VisualFormat', 'NonPrimaryGenre', 'Censor',
'CountryOrigin', 'LanguagesOrigin',
                                      'LanguageFormat', 'Director','Cast', 'Plot'])

    allData = []
    for i in range(df.shape[0]):
        temp = df.iloc[i]
        allData.append(dict(temp))

    cargatemp = {'data': allData}
    return render(request, 'FilmstradamusInicial/carga.html', cargatemp)


def archivotexto(request):
    if os.path.isfile(csp + '\\Data\\ArchivoNuevasPeliculas.csv'):
        os.remove(csp + '\\Data\\ArchivoNuevasPeliculas.csv')
    resultado = ObtenerArchivos.aCargarFinal()
    ObtenerArchivos.archivotexto(resultado)

    Rutinas.prediccion()

    predicciones = pd.read_csv('Data/Prediccion_agregados.csv')

    preds = ObtenerArchivos.conexiongoogle("Predicciones", predicciones)
    allData9 = []
    for i in range(preds.shape[0]):
        temp = preds.iloc[i]
        allData9.append(dict(temp))


    ObtenerArchivos.conexiongoogle("Predicciones", predicciones, 'a')

    allData = []
    for i in range(predicciones.shape[0]):
        temp = predicciones.iloc[i]
        allData.append(dict(temp))



    resultado2 = {'resultado2': allData, 'hist':allData9}
    return render(request, 'FilmstradamusInicial/estimaciones.html', resultado2)


def busqueda(request):
    idmovie = request.POST.get('idpelicula')
    idmovie = idmovie[2:len(idmovie)]
    titulo, presupuesto, duracion, director, sinopsis, actores =
ObtenerArchivos.IMDB(idmovie)
    return
HttpResponse(json.dumps({'titulo':titulo,'budget':presupuesto,'runtime':duracion,'directStr
':director,'sinopsis':sinopsis,'actores':actores}), content_type="application/json")
```

ANEXO II

Código Machine Learning: Rutinas y funciones

A continuación vamos a presentar algunas de las funciones que hemos creado para pre-procesamiento de datos, entrenamiento del modelo y creación de predicciones.

Combinar tablas:

Como ya hemos comentado en la documentación anterior, la primera rutina que hemos creado se encarga de crear las tablas con toda la información necesaria para el procesamiento a nivel cine.

Para ello hemos de llamar a la función `merge_cine`, que a su vez combina los campos de dos tablas distintas:

1. La tabla extraída directamente de Comscore a nivel cine,
2. La tabla que hemos denominado Raíz y que contiene la información ampliada de películas obtenida mediante web scrapping: director, cast, plot, budget, runtime, etc.

```
def merge_cine(cine_ID,
               data_raiz,
               data_key):
    '''
    Al introducir el identificador de un cine y los DataFrame de las tablas raiz
    y de equivalencias la función crea un DF que combina los datos de facturación
    del cine correspondiente con los datos asociados a las películas que ha proyectado.
    Con columnas:
    ['Title', 'Plot', 'Director', 'Cast', 'Genre',  'NonPrimaryGenre', 'Dist', 'Censor',
    'CountryOrigin', 'LanguagesOrigin', 'Runtime', 'Budget', 'Competition',
    'VisualFormat', 'LanguageFormat', 'DayofYear', 'Territory', 'OpWkndAdm', 'OpWkndGross',
    'TotalAdm', 'TotalGross']

    Exporta el DF a un archivo 'Data/MergeCine/cine_IDMerge.csv' y devuelve los campos
    [cine_name, cine_ID, cine_country, number_movies]

    Parámetros
    ----------
    cine_ID :  int
        Identificador de cine que coincide con el valor 'RtkTheatreNumber'.
    data_raiz : pandas DataFrame
        DataFrame que contiene los datos agregados de los cines en la región.
    data_key : pandas DataFrame
        DataFrame que contiene la tabla de equivalencias con las columnas renombradas.
```

```
    '''
    ID = str(cine_ID)

    #Cargamos el DF a nivel cine
    data_cine = pd.read_excel ('../Data/archivos_por_cine/'+ID+'.xls',skiprows = range(0,
3),
    encoding= 'utf-8')
    data_cine.columns = data_cine.columns.str.replace('\n', '').str.replace(' ', '')
                                                                .str.replace('-','')
    data_cine['DayofYear'] = data_cine['WeekofFirstEngagement'].dt.dayofyear
    #dt = atributo del módulo datetime de Pandas.

    #Seleccionamos las columnas de data_cine con las que vamos a trabajar.
    data = data_cine[['Title','VisualFormat','LanguageFormat', 'Dist', 'DayofYear',
                      'WeekofFirstEngagement', 'OpeningWeekendAdm', 'OpeningWeekendGross$',
                      'DateRangeAdmissions', 'DateRangeGross$']]

    #Obtenemos los datos cine_ID, cine_country
    cine_name = data_cine['ScreenID'][0]
    cine_country = data_key.set_index('RtkTheatreNumber').Country.to_dict()[cine_ID]

    #Filtramos por país el DF data_raiz y seleccionamos las columnas que nos interesan.
    data_raiz = data_raiz[data_raiz.Territory == cine_country]
    data_raiz = data_raiz[['USTitle', 'Genre', 'NonPrimaryGenre', 'LanguagesOrigin',
                           'CountryOrigin', 'Censor', 'Runtime', 'Director', 'Budget',
                           'Competition', 'Cast', 'Plot', 'Territory']]

    #Combinamos los DF data_raiz y data_cine via USTitle.
    data = data_cine.merge(data_raiz, how='left', left_on='Title', right_on='USTitle')

    #Deshechamos aquellas entradas que no cuentan con correspondiente USTitle en data_raiz
     (pues se trata de Special Events)
    # y seleccionamos aquellas columnas que nos interesan evitando duplicidades.
    data = data.dropna(axis=0, subset=['USTitle'])
    data = data[['Title', 'Plot', 'Director', 'Cast', 'Genre',  'NonPrimaryGenre', 'Dist',
                 'Censor', 'CountryOrigin', 'LanguagesOrigin', 'Runtime', 'Budget’,
                 'Competition', 'VisualFormat', 'LanguageFormat', 'DayofYear',
                 'WeekofFirstEngagement', 'Territory', 'OpeningWeekendAdm',
```

```
            'OpeningWeekendGross$', 'DateRangeAdmissions', 'DateRangeGross$']]

    #Renombramos las columnas
    data = data.rename(columns = {'OpeningWeekendAdm':'OpWkndAdm',
                                  'OpeningWeekendGross$':'OpWkndGross',
                                  'DateRangeAdmissions':'TotalAdm',
                                  'DateRangeGross$':'TotalGross'})

    #Eliminamos Special Events que no corresponden a películas 'programables'
    data = data[data.Genre!='Special Events']
    data = data[data.Genre!='Rock/Pop Concert']
    data = data[data.Genre!='Classical Concert']

    #Resteamos el índice.
    data.reset_index(inplace = True, drop = True) #la cláusula drop=True elimina la columna
                                                  # del índice, podemos usarla más tarde.
    number_movies = len(data.index)

    #Exportamos el csv
    data.to_csv('../Data/MergeCine/'+ID+'Merge.csv', index=False, encoding= 'utf-8')

    return([cine_ID, cine_name, cine_country, number_movies])
```

Llamaremos a esta función de forma iterativa, recorriendo un listado de cines que se encuentra en una tabla externa, cargando todos los archivos csv correspondientes a los cines de dicha tabla [en total unos 150, el número puede variar en el tiempo si se abren o cierran cines en la región]. Además de crear y exportar los Pandas Data Frames combinados, la función nos permite crear una tabla extra que recoge información útil sobre el cine como el número de películas que se exhibieron en el periodo que vamos a analizar. De cara a entrenar el modelo y crear predicciones nos interesa poder fijar un umbral mínimo en el número de películas que se visionaron.

Funciones para preprocesar las columnas categóricas:

Tal y como hemos explicado en el documento principal, tras un análisis detallado de las variables categóricas del modelo hemos llegado a la conclusión de que hemos de aplicar distintos enfoques a la hora de codificarlas para poder entrenar el modelo.

Con lo que hemos denominado variables categóricas simples (Distributor, Country, Language, Genre y Censor) utilizamos la siguiente función, que nos permite realizar un One-Hot encoding generando una categoría extra a la que denominamos Otros (donde incluímos aquellas categorías que no alcanzan un umbral mínimo de aparición en nuestros datos).

```
def OH_other(data,
             factor,
             factor_pct,
             other_name):
    '''
    Genera columnas tipo One-hot encoding con aquellas categorías que
    aparezcan en la columna 'column' al menos 'thresh' veces.

    Parámetros
    ----------
    data :  pandas DataFrame
        Tabla de datos origen.
    factor : str
        Nombre de la columna factor.
    factor_pct : int
        Porcentaje de 0 a 100 mínimo de observaciones del valor del factor para
        ser graficado.
    other_name : str
        Nombre de la columna para la categoría Other del OH-encoding.
    '''
    OH = pd.get_dummies(data[factor])
    factor_other = pd.value_counts(data[factor]) < (data[factor].count()*factor_pct/100)
    if factor_other.sum() == 0:
        return OH
    else:
        return OH.loc[:, ~factor_other].join(OH.loc[:,
                        factor_other].sum(1).rename(other_name))
```

En el caso de las variables Cast y Director hemos explotado dos enfoques completamente distintos, pero que hemos visto que se complementan.

En primer lugar, mediante la función `TestScoringCast` se codifica cada elenco y director con una cifra de 0 a 99 que se obtiene al realizar la media aritmética de los valores de cada uno de los miembros en un ránking previo (que se ha guardado en los archivos `Director.csv` y `Casting.csv`) y que se obtiene a partir de las recaudaciones de las películas en las que han participado dichos actores/directores.

```
def TestScoringCast(data,
                    territory,
                    defscore,
                    var_objective):
    '''
    Añade dos columnas ('Director_Ranking' y 'Cast_Ranking') al dataframe 'data' que
    contienen la media de los valores del ránking (0 a 99) de los actores/directores que
    participan en cada película. Si un actor o director no aparece en el ránking, su
    contribución a la media será el valor fijado defscore.

    Parámetros
    ----------
    data :  pandas DataFrame
        Tabla de datos para calcular el ranking (data_train).
    territory : str
        Código del territorio en el que está localizado el cine [GT, HN, SV, CR, NI, PA].
    def_score : int
        Número de 0 a 99 para completar los valores de directores/actores que no estén en
el
    ranking raíz.
    var_objective : str
        Nombre de la columna de la categoría en la que se quiere realizar el ránking.
    '''

    fields=('Director', 'Cast')
    for f in fields:
        if os.path.exists("../Data/"+f+'.csv'):
            datatxt = pd.read_csv("../Data/"+f+'.csv', index_col= False, sep=',',
    header=None, encoding = 'utf-8')
            datatxt.columns = ['Territory','Name','Objetivo','Score']
        if f == 'Director':
            df = data.loc[:,('Territory','Director')]
```

```
else:
    df = data.loc[:,('Territory','Cast')]

df = df.loc[(df['Territory']==territory)]
if f == 'Director':
    df = df.assign(Director=df[f].str.split(',')).explode(f)
else:
    df = df.assign(Cast=df[f].str.split(',')).explode(f)

new_df= df.drop_duplicates(subset=['Territory',f], keep="first")

new_df2 = new_df.loc[(new_df['Territory']==territory)]
datatxt2 = datatxt.loc[(datatxt['Territory']==territory)]

new_df2.index = pd.RangeIndex(len(new_df2.index))

for index, row in data.iterrows():
    if row['Territory']==territory:
        sum=0
        x=0
        for i in (row[f].split(',')):
            found = datatxt2.Name[datatxt2.Name == i].count()
            if found == 1:
                curscore=datatxt2.loc[datatxt2['Name'] == i, 'Score'].item()
                ind = new_df2[new_df2[f]==i].index.item()
                new_df2.loc[ind,f+'_Ranking'] = round(curscore)
                sum=sum + curscore
                x=x+1
            else:
                new_df2.loc[index,f+'_Ranking'] = round(defscore)
                sum=sum + defscore
                x=x+1
        AvgAdm=(sum/x)
        data.loc[index,f+'_Ranking'] = round(AvgAdm)
```

En segundo lugar, con las columnas Cast y Director, realizamos un One-Hot encoding tradicional (que genera un número poco manejable de columnas y añade poca información al modelo) y

después aplicamos PCA para reducir la dimensionalidad de esta matriz. La información la guardamos después en un joblib con el código de cine para poder utilizarla en la predicción.

```
def pca_model(cine_ID,
              data,
              factor,
              componentes):
   '''
```

Al introducir un DF (data_train o data_test) genera otro Pandas DataFrame (X) con la reducción de dimensionalidad según lo indicado en el parámetro componentes. La función genera un modelo que guarda todos los parámetros necesarios en formato joblib, para utilizar después en la predicción futura.

```
   Parámetros
   ----------
   cine_ID :  int
       Identificador de cine que coincide con el valor 'RtkTheatreNumber'.
   data :  pandas DataFrame
       data_test o data_train.
   factor : str
       Nombre de la columna factor.
   componentes : int
       Número entero que indica el número de dimensiones principales a las cuales se desea
reducir
   '''

   #Realizamos la reducción de Componentes Principales
   pca = PCA(n_components=componentes)
   pca_data = pca.fit_transform(data)

   #Convertimos en un DF el resultado del PCA
   pca_data = pd.DataFrame(pca_data)

   #Copiamos el índice del DF original al DF del PCA
   pca_data.index = data.index

   #Definimos los nombres de los campos generados por el PCA
   for n in pca_data.columns.values:
```

```
        pca_data.rename(columns={n:"PCA_" + factor + "_" + str(n+1)},inplace='True')


    #Guardamos el modelo PCA en el disco
    ID = str(cine_ID)
    pca_model =  ID + 'ModelPCA_' + factor + '.joblib'
    model_dir = os.path.join(models_dir, pca_model)
    dump(pca, model_dir)


    return (pca_data)
```

Por último, tenemos las variables categóricas `Title` y `Plot` que claramente son de una naturaleza diferente. Vamos a explotar las posibilidades del procesamiento del lenguaje natural (NLP) para crear 10 categorías que agrupen las películas según los "cercanos" que son sus argumentos.

Title	0	1	2	3	4	5	6	7	8	9
Avengers: Endgame	-0.182544	0.394168	-0.660927	-0.395264	0.712815	-0.012678	-0.688981	0.445523	-0.076842	-0.034290
Avengers: Infinity War	-0.179051	0.357710	-0.691594	-0.430140	0.739142	-0.068913	-0.702194	0.447545	-0.124936	-0.080166
Toy Story 4	-0.246373	0.338860	-0.662842	-0.394565	0.677668	-0.033494	-0.638148	0.424127	-0.085498	-0.048235
Lion King, The	-0.213687	0.381338	-0.683729	-0.403508	0.682616	-0.048827	-0.666961	0.429478	-0.118233	-0.041785
Joker	-0.200666	0.338981	-0.690573	-0.380443	0.697953	-0.034985	-0.682940	0.435740	-0.077097	-0.045753
...	...	...	...	...	...	...	...	...	...	...
Girasoles de Nicaragua	-0.228997	0.384637	-0.731028	-0.432223	0.703339	0.007265	-0.645008	0.413456	-0.122815	-0.014970
Mujer fantastica, Una	-0.209004	0.325873	-0.686193	-0.347946	0.690336	-0.048984	-0.696980	0.480372	-0.149243	-0.092661
1,2,3 A Bailar	-0.292658	0.372217	-0.728460	-0.459043	0.652548	0.071110	-0.652169	0.400932	-0.197942	0.082953
Martian, The	-0.174038	0.388278	-0.709026	-0.390973	0.721584	-0.045986	-0.697906	0.436984	-0.102764	-0.049763
Bridge Of Spies	-0.233628	0.371630	-0.676073	-0.416948	0.679472	-0.037273	-0.654510	0.441350	-0.093692	-0.066481

1107 rows × 10 columns

Fig. 1 Visualización de los valores obtenidos mediante Word2Vec para cada película

Hemos decidido emplear la librería gensin que implementa Word2vec, un algoritmo que permite asignar a cada palabra un vector (en nuestro caso de 10 componentes) que recoge información semántica de la palabra y la relaciona con palabras similares.

```
def NLP_Plot(df,
             window,
             n_clust):
   '''
   Al introducir el Data Frame con la tabla raíz, genera una columna en el DF que contiene
el
```

```
número del cluster al que pertenece la película tras el análisis con word2vec.

Parámetros
----------
data :  pandas DataFrame
    tabla raíz
dim : int
    Número de dimensiones en word2vec
n_clust : int
    Número de clusters a calcular
   '''

 #Seleccionamos las columnas que queremos analizar
 df_pc = df[['Title','Plot']]

 # Creamos el corpus para entrenar el modelo uniendo los plots, una vez limpios.
 df_pc['Plot_limpio'] = df_pc.apply(lambda row: nltk.sent_tokenize(row['Plot_limpio']),
                                    axis=1)
 corpus = reduce(lambda s1, s2: s1+" "+s2, df_pc["Plot_limpio"])

 # Entrenamos el modelo con Word2Vec
 model = word2vec.Word2Vec(corpus[0], size=dim, window=5)

 # Calculamos, para cada plot, la media de los valores de sus palabras.
 df_pc['w2v_Plot'] = df_pc.apply(lambda row: document_vectorizer(row['Plot_limpio'],
                                    model), axis=1)
 X_Plot = df_pc['w2v_Plot']
 X_Plot = pd.DataFrame(X_Plot.apply(lambda x: x[0].tolist()).to_list(),
                       index=df_pc['Title'])

 # Creamos los clusters con KMeans
 km = KMeans(n_clusters=n_clust)
 km.fit(X_Plot)

 clusters = km.labels_.tolist()

 # Generamos una columna 'Cluster' que contiene un número de 0 a 9 que denota el cluster
 # en el que se encuadra la película.
```

```
df['Cluster'] = clusters

return(df)
```

Hemos creado una función, `encoding_total`, que combina todas las funciones anteriores de forma que al introducir un DataFrame genera otro DF con las características predictivas codificadas a partir de los parámetros indicados (entre los parámetros a escoger se encuentra el porcentaje para el umbral del `OH_other`, el país en el que se sitúa el cine, o el número de componentes para PCA).

Dicha función también se encarga de procesar las variables numéricas, que tienen un tratamiento mucho más sencillo y que se resume en cambiar los nulos por valores (la mediana en el caso de Budget, o 0 en otro caso).

Esta función se llamará de forma recursiva sobre el listado de cines que se va a analizar, dejando fuera aquellos de los que tenemos un registro demasiado pequeño [se ha escogido fijar el umbral en 500 películas].

Como ya hemos explicado anteriormente vamos a entrenar un modelo para cada cine, que identificaremos por su ID Comscore. En esta fase hemos hecho un grid search para poder escoger el número de validaciones cruzadas y el modelo para realizar el entrenamiento.

De entre las posibles métricas, hemos optimizado con respecto a RMSE, ya que para nosotros es de vital importancia predecir los grandes estrenos (outliers en el modelo por varios órdenes de magnitud).

Después de varios estudios preliminares nos hemos decantado por 10-fold CV con Random Forest con los siguientes parámetros:

```
RandomForestRegressor [bootstrap=False, criterion=mse,
                       max_depth=2, random_state=123,
                       verbose = 10, max_features='sqrt']
```

Tras entrenar los modelos, extraer las predicciones y recoger los datos para el análisis, se exportan los modelos para cada cine en formato joblib para poder realizar las predicciones de las películas a estrenar.

Predicción a partir de los modelos creados:

Por último vamos a mostrar cómo generamos las predicciones a partir de una rutina que recorre los cines que nos interesan, importando los modelos pre-entrenados para finalmente exportar todos los datos necesarios para el análisis.

```
def prediccion():
    '''
    Toma los datos de la tabla metida a mano '../Data/ArchivoNuevasPeliculas.csv' en la web
    (con los mismos campos que tienen IDMerge.csv menos los Adm&Gross,
    y sin DayofYear (con fecha de estreno) ni Competition)
    Y genera una predicción para cada cine en los países que se va a estrenar dicha
película.
    Calcula Competition y DayofYear
    Genera: Tabla con [Película,  País, Predicción]
            Tabla con Película por pais con datos por cine y agregados.

    '''

    # Cargamos el archivo con los datos metidos a mano desde la web y el listado de cines.
    X_pred = pd.read_csv('../Data/ArchivoNuevasPeliculas.csv')
    cines = pd.read_csv('../Data/PrediccionCine/Cines.csv', encoding= 'utf-8')

    # Añadimos la columna DayofYear al dataset.
    X_pred['DayofYear'] = pd.to_datetime(X_pred['WeekofFirstEngagement'],
                                         format = '%Y-%m-%d').dt.dayofyear

    X_pred['Competition'] = 1 #Fijamos el valor a falta de la función de Mario

    aux = []

    #Recorremos los países para obtener las predicciones
    for t in X_pred['Territory'].unique():
        data_t = X_pred[X_pred.Territory == t] #Seleccionamos las películas en cada país
        data_t['Prediccion'] = 0
        #Recorremos los cines de dicho país
        for [idx, num, country] in zip(cines['Cine_ID'].to_list(),
                                       cines['Number_movies'].to_list(),
```

```
                                cines['Country'].to_list()):
        if (num >=500) and (country == t):
            idx = str(idx)
            med_budget = cines[cines.Cine_ID == idx ]['Med_Budget']
            #Realizamos el encoding
            X_t = encode_pred(data_t, med_budget, 0, t)
            #Recuperamos el modelo entrenado
            model_name =  idx + 'ModelRF.joblib'
            model_dir = os.path.join(models_dir, model_name)
            model = load(model_dir)
            #Tenemos que hacer compatibles las columnas de los encodings
            f_names = model.feature_names
            X_t = col_consistency(f_names, X_t)
            pred = model.predict(X_t)
            #Añadimos la predicción por cine
            data_t[idx] = pred
            #Añadimos la predicción acumulada
            data_t['Prediccion'] = data_t['Prediccion'] + pred
    #Guardamos las predicciones por cine en una tabla para cada país.
    data_t['Prediccion'] = data_t['Prediccion'].round(0).astype(int)
    data_t.to_csv('../Data/Prediccion_'+t+'.csv', index=False, encoding= 'utf-8')
    #Guardamos los datos agregados para crear la tabla final
    aux.append(data_t[['Territory', 'Title', 'Genre', 'WeekofFirstEngagement',
                       'Prediccion']])

#Creamos un Dataframe con los datos ['Title', 'Territory', 'Prediccion']
X = pd.concat(aux, axis=0)
X['Prediccion'] = X['Prediccion'].round(0).astype(int)
X.to_csv('../Data/Prediccion_agregados.csv', index=False, encoding= 'utf-8')
```

Anexo III

Instrumentos de visualización y cuadros de mando

CUADRO GENERAL DEL CINE EN CENTRO AMERICA

COMPARACIÓN REAL VS PREDICCIÓN MODELO

CUADRO DE LA INDUSTRIA SEGÚN LAS PREDICCIONES DEL MODELO

(Indicadores Cinemark, Películas Top)

<u>CUADRO DETALLE DEL SHAREPOINT SEGÚN PREDICCION</u>

(por Género, por película)

CUADRO DE CARCATERÍSTICAS MÁS IMPORTANTES DEL MODELO