PROYECTO 1:

**Smart\_Hours.**

Integrantes del grupo: Pau Albiol, David Álvarez, Guillermo Creus

Día de clase: 29/05/2018

Mesa: 4

Índice:

1)Objetivos (página 3)

2)Alcance (página 3)

3)Análisis del problema y descripción de la solución (página 4)

4)Herramientas e implementación (Página 5)

5)Planificación y costes (página 6)

6)Resultados y conclusión (página 6)

7)Bibliografía (página 6)

1)Objetivos:

Creación de un horario de estudio que se adapte a los objetivos académicos del cliente.

2)Alcance:

Funcionalidades:

Smart\_Hours genera un horario y ofrece cuatro tablas y un gráfico que contienen datos de interés enfocados al cliente. Las horas del horario están distribuidas de manera adecuada para que cliente pueda cumplir con sus objetivos académicos.

Para crear el horario, el cliente debe interactuar con la web rellenando tres formularios.

1. El primer formulario, consiste en indicar qué horas de la semana (de lunes a viernes, de 9:00 h a 23:00 h) tiene ocupadas el cliente para que no se asignen horas de estudio en dichas horas no disponibles.
2. En el segundo, éste debe elegir su rendimiento e introducir las asignaturas que quiere en su horario. El parámetro del rendimiento indica “el objetivo”. Hay cuatro opciones a elegir:
3. Mínimo: Aprobar con 5 las asignaturas elegidas.
4. Medio: Mantener la nota media del cliente o aprobar con 5 en caso de que esta media sea de suspenso.
5. Alto: Aumentar la nota media en un 10% o aprobar con 5.5 en caso de que esta media sea de suspenso.
6. Manual: El cliente puede introducir qué notas quiere para cada asignatura.
7. En el tercer formulario se pide la nota de selectividad y las diez asignaturas de primero (es el Historial académico del cliente).

Este parámetro permite predecir las notas que va a sacar el cliente y calcular cuán lejos está de sus objetivos.

Finalmente se generará el horario y se mostrarán las distintas tablas. Estas son:

1. Fin de semana:

Tabla que indica las horas a dedicar durante el fin de semana si las horas de lunes a viernes son insuficientes para que el cliente cumpla sus objetivos.

1. Notas:

Tabla donde de cada asignatura elegida, se da información sobre la nota predicha, la nota deseada (es decir el objetivo) y la media global.

1. Distribución de horas de estudio:

Gráfico tipo *3d-Pie* que representa en porcentajes, el tiempo de estudio de cada asignatura (respecto al total de horas de estudio).

1. Estudiantes con notas más cercanas:

Tabla que compara las notas del Historial académico del cliente con el de los doce estudiantes más cercanos a él.

1. Notas de las asignaturas matriculadas de los estudiantes más cercanos:

Tabla que compara las notas predichas del cliente frente a las notas de los doce estudiantes más cercanos a él.

Otras funcionalidades (más técnicas):

El horario no muestra todas las horas si estas están vacías. Es decir, si el cliente estudia a partir de las 18:00 h, el horario no mostrará las horas vacías de 9:00 h a 18:00 h.

Por comodidad, para rellenar los *chechboxs* en el apartado de selección de Horas ocupadas, es posible clicar y arrastrar con el ratón haciendo menos tedioso este formulario.

Condiciones y limitaciones:

Como mínimo el cliente debe introducir una nota en el parámetro de Historial académico. Sólo se aceptan caracteres numéricos. En caso de decimales se debe indicar con un punto. (Por ejemplo: 7.5)

La tabla del horario abarca de lunes a viernes, de 9:00 h a 23:00 h. Si el número de horas de estudio de lunes a viernes es insuficiente, las horas restantes se asignan en el fin de semana.

3)Análisis del problema y descripción de la solución:

Descripción de los problemas:

1. Problema del filtraje:

Se ha partido de tres base de datos que contenían resultados académicos de estudiantes del ETSEIB. La primera de ellas contenía las notas de selectividad, la segunda las notas de la fase inicial y la tercera las notas de la fase no inicial (hasta Q1 17/18). Debido que no siempre hay los mismos estudiantes, hay más datos de los necesarios y puede que un estudiante tenga múltiples notas de la misma asignatura (puede ser repetidor) se procedido a un filtraje de datos.

1. Problema de predicción:

Dado que un factor para calcular las horas de estudio es la nota predicha del cliente, obviamente se necesitan realizar predicciones (de una variable numérica continua). Para ello se ha utilizado el método de K-Vecinos más cercanos.

1. Problema de asignación de horas:

Una vez predichas las notas, se deben comparar con los objetivos del cliente y traducir a “horas de estudio”.

1. Problema de ordenación de horas:

Obtenidas las horas de estudio para cada asignatura, estas se deben distribuir en un horario de forma ordenada y lógica (evitar problemas del tipo: en un día hay que estudiar 4 horas seguidas de la misma asignatura).

Solución propuesta (Planteamiento global de funcionamiento):  
……….

4)Herramientas e implementación:

Herramientas de soft usadas:

Para el filtrado de datos se ha usado pandas, una librería de Python destinada al análisis de datos que proporciona unas estructuras de datos flexibles y que permiten trabajar con ellos de forma eficiente. Más concretamente se ha trabajado con DataFrame (los datos se estructuran en forma de tablas).

El programa se ha desarrollado en Python…

Para crear la aplicación web se ha usado Flask. Flask es un framework minimalista escrito en Python que permite crear aplicaciones web con un mínimo número de líneas de código (basado en el motor de templates Jinja2).

La página web funciona con archivos html y javascript…

Para trabajar en equipo de manera eficiente se creó un repositorio público en GitHub (<https://github.com/David-98/Horario>) . Github es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el controlador de versiones Git.

Descripción de la solución concreta adaptada:

5)Planificación y costes:

Para realizar el proyecto de Smart\_Hours, una de las primeras cosas que se creó fue un plan de ruta cuya planificación abarcaba cerca a dos meses de desarrollo. Las horas dedicadas en Smart\_Hours han sido para cada integrante:

* 5 horas de clase de teoría dónde se aprendió sobre el uso de pandas y flask.
* Un promedio de 55 horas de autoaprendizaje

Es decir, en total unas 180 horas.

El único coste del proyecto ha sido el consumo energético para alimentar los ordenadores de los integrantes.

Suponiendo que un ordenador usa 100W por 1 hora diaria a 0,12 € el kWh, calculando para un integrante el coste mensual es de 0,43 € al mes. Dado que el proyecto ha durado 2 meses y Smart\_Hours ha sido desarrollado por 3 integrantes, el coste energético estimado es de **2,6 €**.

6)Resultados y conclusiones:

Para realizar este proyecto, una buena comunicación y coordinación entre el equipo ha sido imprescindible. La estrategia de haber dividido y planificado el proyecto en pequeños problemas y asignar roles a cada integrante ha resultado muy eficiente. Ha simplificado las cosas pues cada integrante se focalizaba en un problema y en caso de no saber continuar, entre todos se ha buscado una solución o propuesto alternativas para solucionar (o en algún caso evitar) el problema.

Un gran impacto de este proyecto ha sido el de autoaprendizaje pues siempre que aparecía algún inconveniente, en internet se hallaba la solución.

También se han adquirido competencias genéricas sobre la comunicación oral y escrita. Ha habido una clara evolución en cómo se han realizado las presentaciones orales y el estilo de los PDF’s que se proyectaban.

7)Bibliografía:

<https://stackoverflow.com/>

<https://www.w3schools.com/>

<https://pandas.pydata.org/pandas-docs/stable/10min.html>