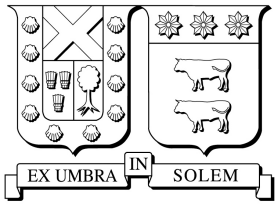

Análisis Inteligente de Datos - Tarea I

Prof. Ricardo Ñanculef
jnancu@inf.utfsm.cl

Instrucciones

- El objetivo de estas tareas/talleres es que pongan en práctica la teoría y los métodos discutidos en clases. El énfasis debe ser puesto por lo tanto en: la comprensión del problema abordado, la implementación computacional de los métodos correspondientes, las decisiones prácticas que haya tenido que tomar para completar el análisis y la interpretación de los resultados.
- Como hemos acordado en clases, lo ideal es que hagan esta tarea en parejas. Cada miembro del equipo debe estar en condiciones de exponer cada punto del trabajo realizado.
- Se recomienda el uso de Python como lenguaje de programación, junto a todo el ecosistema de librerías que hemos mencionado en clases, incluyendo: pandas, numpy, sklearn, statsmodels y seaborn.
- La realización de la tarea consiste en preparar un breve informe y presentación (slides) del trabajo realizado. Por favor, incluya en estos productos sólo aquello esencial para dar a entender lo realizado. Los programas utilizados contendrán todos los detalles. Éstos deben ser entregados y mantenidos en un repositorio público como github.
- La fecha entrega de todo el material entregable será siempre el día en que los resultados se discutirán en clases y será acordada oportunamente.
- El formato de entrega será electrónico vía email a jnancu@inf.utfsm.cl.

1 Manipulación Básica de Dataframes y Visualización I

Kaggle [1] es una plataforma que permite organizar "competencias" de minería y análisis predictivo de datos, en la cual una empresa o grupo de investigación define un problema y científicos de datos de todo el mundo compiten por resolverlo, de acuerdo a un criterio (scoring function) previamente definido. Generalmente existe una recompensa en dinero para la o las mejores soluciones. En esta sección trabajaremos con los datos publicados para una competencia *Kaggle* titulada *Titanic: Machine Learning from Disaster* [2].

El 15 de abril de 1912, un trasatlántico británico, denominado RMS TITANIC, se hundió durante su viaje inaugural desde Southampton a Nueva York, después de chocar con un iceberg, causando la muerte de 1502 personas (de un total de 2224 pasajeros). El hundimiento del TITANIC es uno de los naufragios más famosos de la historia. El análisis de datos que se llevará a cabo en esta sección será guiado por la siguiente pregunta: ¿Qué pasajeros tenían mayor probabilidad de sobrevivir? ¿Es posible predecir quién se salvará y quién no?

Los datos se encontrarán disponibles en formato "csv" (comma separated values) en moodle, separados en dos subconjuntos: *titanic-train.csv* y *titanic-test.csv*. A no ser que se indique lo contrario, se debe utilizar

sólo el primer subconjunto de datos. Como comprobará, se le darán indicaciones bastante precisas sobre los comandos a utilizar si decide trabajar en el ecosistema computacional sugerido[1].

(a) Utilizando la librería *pandas*, construya un dataframe con los datos a analizar. Comandos útiles:

```
import pandas as pd
pd.options.display.mpl_style = 'default'
data = pd.read_csv('tarea1/data/titanic-train.csv')
```

El archivo de entrada ha sido modificado para que el comando anterior no funcione!. Para poder cargar exitosamente los datos tendrá que indicar a la función *read_csv* que las variables correspondientes a cada ítem se encuentran separadas por ';' y no por ','.

(b) Determine de cuántos ítems y de cuántas variables se dispone para hacer el análisis. Determine el nombre de cada variable y describa la semántica asociada a cada una, a partir de la información publicada en [2]. ¿Hay columnas con datos faltantes? Explore además la información que se obtiene con las funciones *info()* y *describe()* asociadas a un dataframe. Comandos útiles:

```
data.shape
data.info()
data.describe()
```

(c) Eche un vistazo a las primeros ítems del dataframe, a los últimos y a los ítemes 200 a 210. Repita, visualizando sólo las columnas correspondientes a las variables "Sex" and "Survived". Comandos útiles:

```
data.tail()
data.head()
data[100:110][:]
data[['Sex','Survived']].tail()
data[['Sex','Survived']][100:110]
```

(d) Determine qué proporción de hombres y qué proporción de mujeres sobrevivió al naufragio. Construya un gráfico de barras para visualizar esta información. ¿Está correlacionado el género con la probabilidad de sobrevivencia? Determine además la proporción de sobrevivientes que son mujeres y la proporción de sobrevivientes que son hombres. Construya un gráfico de barras para visualizar esta información. Discuta la diferencia con las proporciones calculadas al inicio, si lo que se desea es predecir la sobrevivencia del pasajero. Comandos útiles: Comandos útiles:

```
data['Sex'].value_counts()
data.groupby('Sex').Survived.count()
data.groupby('Sex').Survived.mean()
data.groupby('Survived')['Sex'].value_counts()
data.groupby('Sex')['Survived'].value_counts()
data.groupby('Sex')['Survived'].value_counts().unstack().plot(kind='bar')
grouped_props = data.groupby('Survived')['Sex'].value_counts()/ \
data.groupby('Survived').size()
grouped_props.unstack().plot(kind='bar')
```

(e) Compare las edades de los pasajeros que sobrevivieron y de aquellos que murieron. Parta comparando las edades medias. Construya luego boxplots e histogramas para visualizar mejor esta información. ¿Está correlacionada la edad con la probabilidad de sobrevivencia? Determine si existen datos para los cuales no se conoce la edad (datos faltantes), cuántos son y averigüe cómo influye esto sobre el cálculo de las estadísticas calculadas anteriormente. ¿Qué edad tenía el mayor pasajero abordo? ¿Sobrevivió? Comandos útiles:

[1]Todos los trozos de código que se muestran a continuación suponen su ejecución en el intérprete de Python.

```
data.groupby('Survived')['Age'].mean()
data.boxplot(column='Age',by='Survived')
data.hist(column='Age',by='Survived')
sum(data[data.Survived==0]['Age'].isnull())
sum(data[data.Survived==0]['Age'].notnull())
data[data.Age==data['Age'].max()]
data.hist(column='Age',by='Survived')
data.boxplot(column='Age',by='Survived')
```

Como alternativa, puede construir estos gráficos usando seaborn

```
pd.options.display.mpl_style = None
import seaborn as sns
import matplotlib.pyplot as plt
survived_data = data[data.Age.notnull()][data.Survived==1]
died_data = data[data.Age.notnull()][data.Survived==0]
fig, ax = plt.subplots()
sns.distplot(survived_data['Age'])
sns.distplot(died_data['Age'])
plt.show()
pd.options.display.mpl_style = 'default'
```

(f) Diseñe un mecanismo para imputar las edades faltantes, es decir para reemplazar el valor faltante por un valor que tenga sentido. Por ejemplo, el código que se muestra a continuación imputa una edad faltante usando la mediana de la edad de las mujeres si el pasajero en cuestión es una mujer y la mediana de la edad de los hombres en cualquier otro caso.

```
median_male = data[(data.Age.notnull()) & (data.Sex=='male')]['Age'].median()
median_female = data[(data.Age.notnull()) & (data.Sex=='female')]['Age'].median()
data[(data.Age.isnull()) & (data.Sex=='female')]['Age'] = median_female
data.loc[(data.Age.isnull()) & (data.Sex=='female'), 'Age'] = median_female
```

Note que la tercera instrucción no da como resultado lo que se espera. Explique porqué.

(g) Determine cuántas clases diferentes de pasajeros viajaban abordo. ¿Qué proporción de cada clase sobrevivió al naufragio? ¿Qué proporción de los sobrevivientes corresponde a cada clase? Construya un gráfico de barras que resuma esta información. ¿Está correlacionada la clase del pasajero con la probabilidad de sobrevivencia? Determine ahora, qué porcentaje de las mujeres sobrevivientes era de cada clase, y qué porcentaje de los hombres sobrevivientes era de cada clase. Construya un gráfico de barras que resuma esta información. Comandos útiles:

```
data['Pclass'].unique()
data.groupby(['Pclass']).groups.keys()
data.groupby(['Survived', 'Pclass']).size()/data.groupby(['Survived']).size()
data.groupby(['Pclass','Survived']).size()/data.groupby(['Pclass']).size()
females = data[data.Sex == 'female'].groupby(['Survived','Pclass']).size()/ \
data[data.Sex == 'female'].groupby(['Survived']).size()
females.unstack().plot(kind='bar')
```

(h) En base a los resultados anteriores, diseñe una regla sencilla que permita predecir quién se salvará y quién no considerando el sexo y la clase del pasajero. Agregue una columna al dataframe que corresponda a sus predicciones. Analice la precisión de sus predicciones (proporción de las veces que se predice muerte/sobrevivencia y efectivamente la persona murió/sobrevivió) y el recall de sus predicciones (proporción de los muertos/sobrevivientes que su regla efectivamente predice como tales). Comandos útiles:

```
data['prediction']=0
data.prediction[data.sex == 'female']=1
```

```
data[data.prediction==1][data.Survived==1].size/float(data[data.prediction==1].size)
data[data.prediction==1][data.Survived==1].size/float(data[data.Survived==1].size)
data.to_csv('predicciones-titanic.csv')
```

(i) Construya un boxplot que resuma los valores que toma la variable *Fare* (precio del boleto) en el dataset. Filtre los valores extremos (outliers) y construya un histograma con los valores típicos correspondientes a las personas que se salvaron y a las que no. ¿Está correlacionada esta nueva variable con la probabilidad de sobrevivencia? Construya ahora una nueva variable en el dataframe que corresponda al rango de precio del boleto, usando 5 intervalos de precios: $[0, 9]$, $[10, 19]$, $[20, 29]$, $[29, 39]$, $[40, \infty]$. Modifique la regla de clasificación construida en el punto anterior para tener cuenta de esta nueva variable y analice la precisión y el recall de sus predicciones.

```
data.boxplot(column='Fare')
dataFareTyp=data[data.Fare<50]
dataFareTyp.hist(column='Fare')
pd.options.display.mpl_style = None
import seaborn as sns
dataFareTypSurv=data[(data.Fare<63) & (data.Survived==1)]
dataFareTypDied=data[(data.Fare<63) & (data.Survived==0)]
fig, ax = plt.subplots()
sns.distplot(dataFareTypSurv['Fare'])
sns.distplot(dataFareTypDied['Fare'])
plt.show()
pd.options.display.mpl_style = 'default'
```

(j) Determine la precisión y el recall de la regla de clasificación construida en (h) e (i) sobre el conjunto de datos de pruebas (*titanic-test.csv*).

2 PCA y y Visualización II

En esta sección trabajaremos con datos epidemiológicos obtenidos desde el sitio de la fundación *Gapminder* [3]. El primer dataframe corresponde a la incidencia de la Tubercolosis (número de personas infectadas por cada 100.000 habitantes) en diferentes países del mundo sobre base anual entre 1990 y 2007. El segundo dataframe corresponde a la incidencia del VIH (porcentaje de adultos entre 15 y 49 años infectados) en el mundo sobre base anual entre 1979 y 2011.

La tuberculosis es una enfermedad infecciosa causada por micro-bacterias (fundamentalmente Mycobacterium tuberculosis). Es considerada una de las primeras enfermedades humanas de las que se tiene constancia y sigue siendo hoy en día la enfermedad infecciosa humana más importante que existe en el mundo. Se estima que la enfermedad mata 4400 personas cada día y se estima que un 80% de los casos se concentran en sólo 22 países. Por otro lado, desde su identificación hace más de 30 años, la pandemia del VIH ha infectado a más de 60 millones de personas y causado la muerte de al menos 25 millones a nivel mundial. La presencia del virus está documentada en la mayor parte de los países del planeta, pero las tasas de prevalencia y su evolución varían de país en país.

Los datos se encontrarán disponibles en formato "csv" (comma separated values) en moodle: *TB.csv* y *HIV.csv*. Como en clases hemos discutido detalladamente cómo hacer PCA, no se entregarán "tips" computacionales con respecto a esta parte y se solicita construir las proyecciones "a mano" sin hacer uso de una librería específica para PCA. Es permitido usar librerías para obtener las descomposiciones matriciales necesarias.

(a) Construya un dataframe con los datos a analizar (atención: los datos se encuentran separados por ';' y el punto decimal se encuentran codificado con el símbolo ',' en vez de '.'). Como podrá comprobar, existe un gran número de datos faltantes para los años 1979 a 1989, por lo que restringiremos el análisis a los años 1990-2011. Elimine las columnas correspondientes a los años 1979-1989. De los datos

restantes elimine todas las filas (países) para los que existan datos correspondientes al año 2010. Sobre el resultado obtenido remueva todas las filas con datos faltantes. Debería obtener un dataframe de 139 registros y 22 columnas.

```
import pandas as pd
data = pd.read_csv('tarea1/data/HIV.csv',sep=';',decimal=',', index_col = 0)
data = data.drop(data.columns[range(0,11)], axis=1)
data = data[data['2010'].notnull()]
data = data.dropna()
data.shape
data.index.names = ['country']
data.columns.names = ['year']
```

(b) Construya un gráfico de líneas que compare la evolución de la incidencia del VIH en diferentes países del mundo. Elija arbitrariamente $2-4$ países por continente.

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots(figsize=(8, 4))
data.loc[['Chile','Argentina','Italy','Cameroon', 'Congo, Rep,'],'1990':].T.plot(ax=ax)
ax.legend(loc='upper left', bbox_to_anchor=(0.1, 1.1),prop={'size':'x-small'},ncol=6)
plt.tight_layout(pad=1.5)
plt.show()
```

(c) Prepare la matriz de datos que se utilizará para proyectar las series correspondientes a cada país en sus 2 componentes principales. Pre-procese los datos como hemos discutido en clases, de manera que cada columna tenga media nula y varianza 1.

```
from sklearn.preprocessing import StandardScaler
X = data.ix[:,'1990':'2011'].values
X_std = StandardScaler().fit_transform(X)
```

(d) Calcule las componentes principales de los datos, en orden de importancia. Construya un gráfico de barras que de cuenta de la varianza explicada por las primeras 4. Compute luego la la proyección de las observaciones originales en las primeras dos componentes para obtener una representación bi-dimensional de la serie que corresponde a cada país. Construya un nuevo dataframe con la representación bi-dimensional.

```
data_2d = pd.DataFrame(Y_sklearn)
data_2d.index = data.index
data_2d.columns = ['PC1','PC2']
```

(e) Construya un *scatter plot* de los datos proyectados en dos dimensiones. Colore los puntos utilizando un código de color secuencial (sequential colormap) que de mayor intensidad a los países con mayor incidencia de la enfermedad (en promedio a lo largo de los años). Concluya sobre qué información codifica la primera componente principal. Colore ahora los puntos utilizando un código de color divergente (diverging colormap) que discrimine a los países donde la incidencia tiende a aumentar con los años (en promedio a lo largo de los años) de aquellos en que tiende a disminuir o de aquellos en que se mantiene más o menos constante.

```
row_means = data.mean(axis=1)
row_trends = data.diff(axis=1).mean(axis=1)
data_2d.plot(kind='scatter', x='PC2', y='PC1', figsize=(16,8), c=row_means,cmap='RdBu')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

(f) Modifique el gráfico obtenido en (e) utilizando un *bubble plot* en vez de un *scatter plot*. Discuta.

```
data_2d.plot(kind='scatter', x='PC2', y='PC1', figsize=(16,8), s=10*row_means, \
c=row_means,cmap='RdBu')
```

(g) Evalúe si conviene agregar al gráfico el nombre del país representado. Discuta.

```
fig, ax = plt.subplots(figsize=(16,8))
row_means = data.mean(axis=1)
row_trends = data.diff(axis=1).mean(axis=1)
data_2d.plot(kind='scatter', x='PC2', y='PC1', ax=ax, s=10*row_means, c=row_means, \
cmap='RdBu')
Q3_HIV_world = data.mean(axis=1).quantile(q=0.85)
HIV_country = data.mean(axis=1)
names = data.index
for i, txt in enumerate(names):
        if(HIV_country[i]>Q3_HIV_world):
            ax.annotate(txt, (data_2d.iloc[i].PC2+0.2,data_2d.iloc[i].PC1))

plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

(h) Cargue ahora los datos correspondientes a la Tuberculosis. Prepare la matriz de datos que se utilizará para proyectar las series correspondientes a cada país en sus 2 componentes principales. Procese los datos como hemos discutido en clases, de manera que cada columna tenga media nula y varianza 1.

```
data = pd.read_csv('tarea1/data/TB.csv',sep=',',thousands=',', index_col = 0)
data.index.names = ['country']
data.columns.names = ['year']
X = data.ix[:,'1990':'2007'].values
X_std = StandardScaler().fit_transform(X)
```

(i) Repita (d) a (g) con los datos de la tuberculosis. Determine qué puede concluir del análisis realizado.

References

[1] https://www.kaggle.com/

[2] https://www.kaggle.com/c/titanic

[3] http://www.gapminder.org/

LaTeX