{desafío}
latam_

Hipótesis y Correlación_

Sesión Presencial 1

{desafío}
latam_

Alcances de la lectura asignada

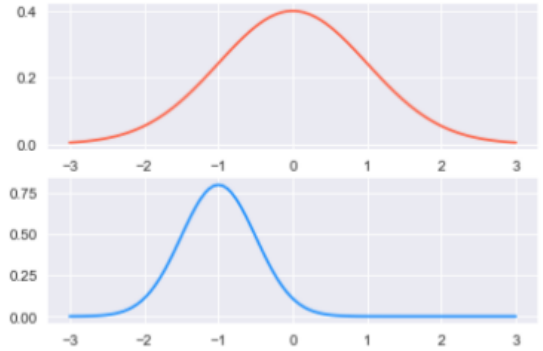
- *Conocer las funcionalidades avanzadas de gráficos estáticos mediante `seaborn`.*
- *Aprender a segmentar datos y los principales criterios de estratificación.*
- *Conocer los principales criterios de transformación de variables.*
- *Aplicar funciones a columnas de datos mediante `ufuncs`, `map-reduce-filter`.*
- *Entender e interpretar la correlación a partir de diagramas de dispersión.*
- *Entender el marco inferencial frecuentista de las hipótesis.*
- *Conocer la distribución t de Student y su aplicación.*
- *Aplicar pruebas de hipótesis simples en el contexto de la inferencia.*

Activación de Conceptos

- En la unidad anterior aprendimos sobre gráficos y algunas distribuciones.
- ¡Pongamos a prueba nuestros conocimientos!

¿Cuál de las dos curvas tiene una menor media?

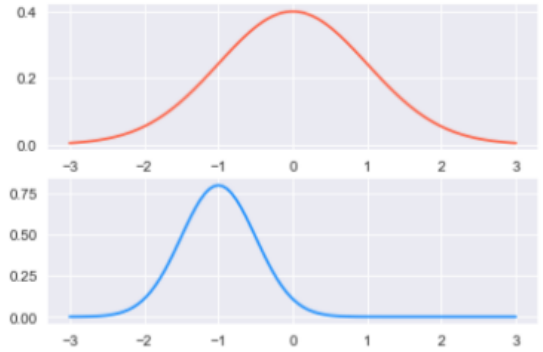
In [2]:
```
plt.subplot(2,1, 1); plt.plot(x_axis, norm_1, color="tomato")
plt.subplot(2,1, 2); plt.plot(x_axis, norm_2, color="dodgerblue");
```



1. Roja
   - Azul
   - Ambas son iguales

¿Cuál de las dos curvas tiene una menor varianza?

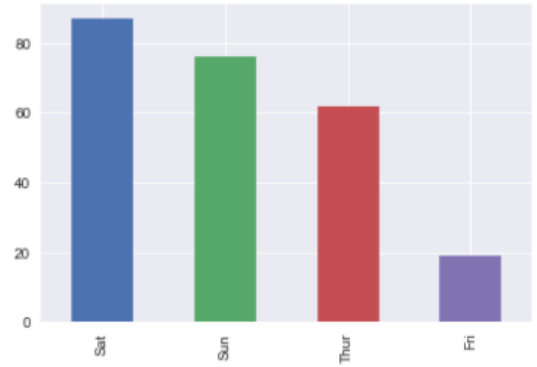
In [3]:
```
plt.subplot(2,1, 1); plt.plot(x_axis, norm_1, color="tomato")
plt.subplot(2,1, 2); plt.plot(x_axis, norm_2, color="dodgerblue");
```



1. Roja
   - Azul
   - Ambas son iguales

¿Cómo podemos replicar el siguiente gráfico?

```
In [5]: question()
```



1. `plt.barplot(count = df['day'])`

- `plt.plot(df['day'], kind='bar')`
- `df['day'].plot(kind='bar)`
- `df['day'].value_counts().plot(kind='bar')`

{desafío}
latam_

¿Cuál de las siguientes frases resume de mejor manera el Teorema del Límite Central?

1. Independiente de la distribución de la variable, la suma y media de las mediciones de cada variable tiende a tener una distribución aproximadamente normal en la medida que $n \xrightarrow{d} \infty$.

- En una sucesión infinita de variables aleatorias i.i.d con expectativa $\mathbb{E}(x_i)$ y varianza σ^2, el promedio de la sucesión convergerá en probabilidad a μ.
- En la medida que el parámetro estimado aumenta, convergerá hacia el parámetro verdadero.
- En la medida que la varianza disminuye, el parámetro estimado convergerá hacia el parámetro verdadero.

Refactorización con seaborn

{desafío}
latam_

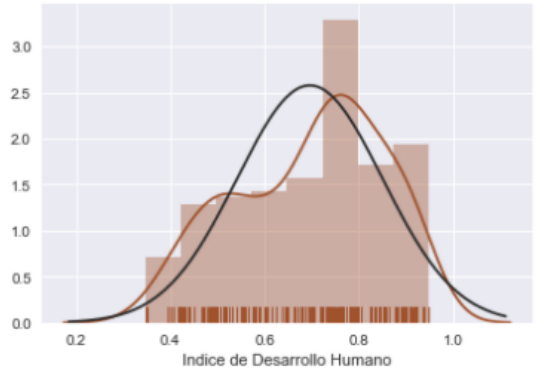
¿Por qué seaborn?

- Resulta que un buen gráfico siempre conlleva escribir grandes piezas de código (si no me creen, vean los scripts de las últimas sesiones).
- `seaborn` busca agilizar el proceso de crear gráficos y sistematizar los protocolos a una serie de elementos.
- A final de cuentas es más probable utilizar los gráficos más comunes (dispersión, histograma, cajas, etc...), que reinventar la rueda.
- Importamos seaborn de la siguiente manera:

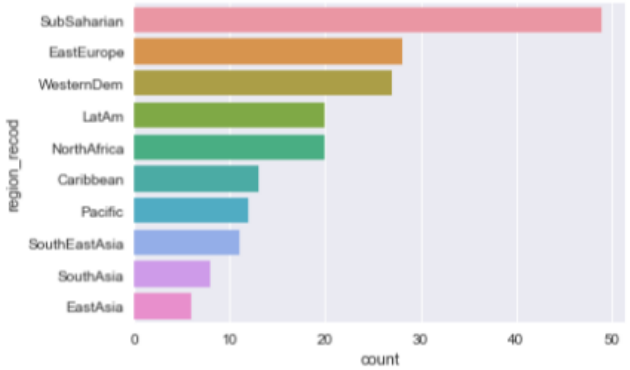
```
import seaborn as sns
```

In [6]:
```
import seaborn as sns
```

In [7]:
```
sns.distplot(df_gob['undp_hdi'].dropna(), rug=True,
             axlabel="Indice de Desarrollo Humano",
             fit=stats.norm, color='sienna');
```

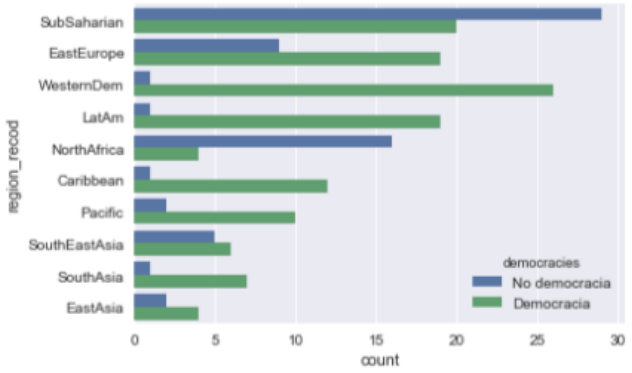


In [9]:
```
sns.countplot(y= df_gob['region_recod'],
              order = df_gob['region_recod'].value_counts().index);
```



In [10]:

```
# generamos una recodificación binaria con np.where
df_gob['democracies'] = np.where(df_gob['gol_inst'] <= 2, 'Democracia', 'No democracia')
sns.countplot(y = df_gob['region_recod'], hue=df_gob['democracies'],
              order = df_gob['region_recod'].value_counts().index);
```



FacetGrid

- Múltiples figuras en un mismo gráfico, que comparten ejes y condicionadas por un valor específico.
- `FacetGrid` busca agilizar el proceso cuando deseamos graficar a lo largo de una serie de valores discretos.
- `plt.subplot` es más flexible que `FacetGrid`

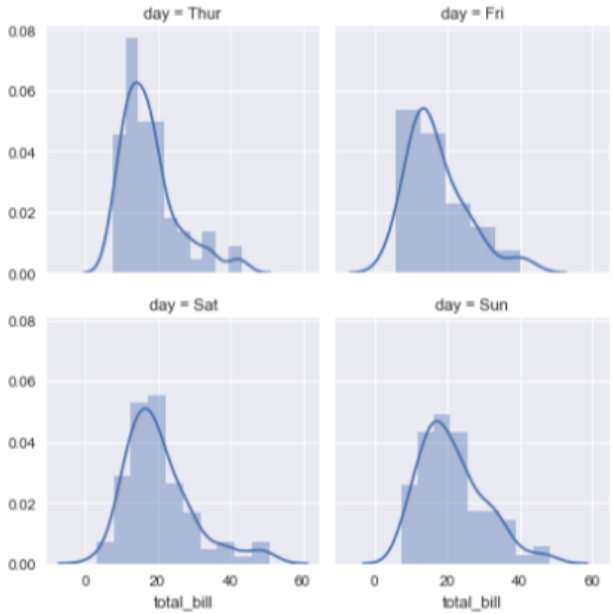
Flujo de trabajo `FacetGrid`

1. Iniciar un objeto `FacetGrid` ⇝ definir DataFrame, Variable y cantidad de columnas.

1. Aplicar gráficos con `map` al objeto creado.

In [11]:

```
grid = sns.FacetGrid(df, col="day", col_wrap=2)
grid = grid.map(sns.distplot, "total_bill")
```



Scatterplots

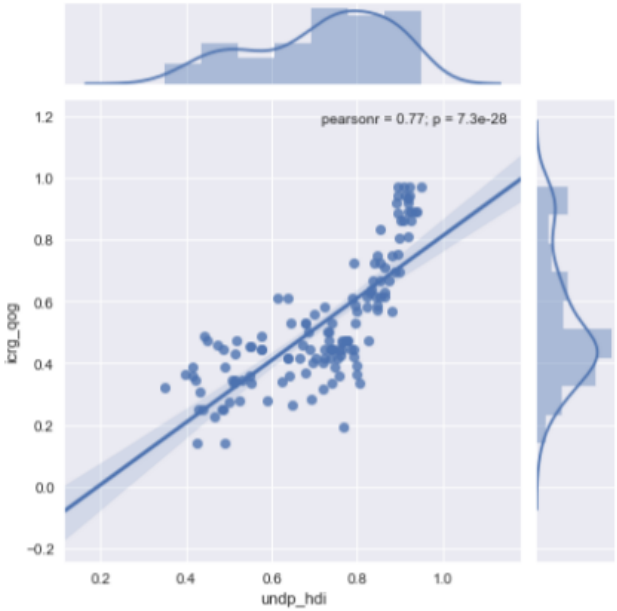
{desafío}
latam_

¿Qué son?

- Permite visualizar observaciones mediante coordenadas cartesianas:
  - Eje X ⇝ Línea Horizontal.
  - Eje Y ⇝ Línea Vertical.

- Informa sobre cómo se comportan dos variables y su posible relación.

In [13]:
```
sns.jointplot(scatter_data['undp_hdi'], scatter_data['icrg_qog'], kind='reg');
```



Correlación y Covarianza

{desafío}
latam_

Definición

- Sólo sirven para cuantificar el grado de asociación entre dos variables.
- **Covarianza:**

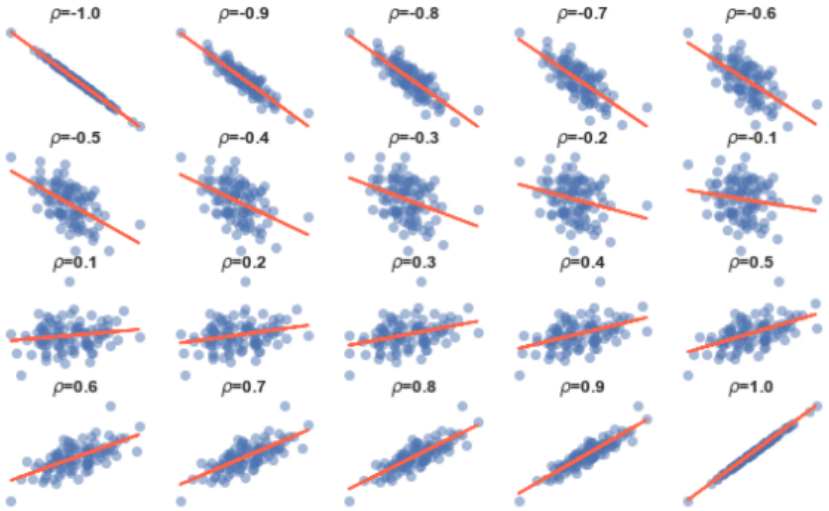
$$\text{Covarianza}(x, y) = \frac{1}{N-1} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$$

- **Correlación:**

$$\text{Correlación}(x, y) = \frac{\text{Covarianza}(x, y)}{\sqrt{\text{Varianza}(x)}\sqrt{\text{Varianza}(\text{Y})}}$$

- La correlación varía entre 0 (ausencia de relación) a 1 (relación perfecta directamente proporcional) o -1 (relación perfecta inversamente proporcional).
- Algunas salvedades:
  - El valor de ρ no depende de las unidades de medición.
  - Tampoco depende de qué variable se denomina x e y

In [15]:
```
plt.figure(figsize=(10, 6));gfx.generate_corr_matrix()
```



PairGrid

Componentes

- `PairGrid` permite realizar cruces bivariados en una serie finita de variables mediante una grilla.

- Ésta se compone de tres partes que se mapean de similar manera a `FacetGrid`:
    - Diagonal Principal `map_diag`
    - Triángulo Inferior `map_lower`
    - Triángulo Superior`map_upper`

In [22]:

```
grid = sns.PairGrid(working_subset, )
grid = grid.map_diag(sns.distplot)
grid = grid.map_lower(sns.kdeplot, cmap="Blues_d")
grid = grid.map_upper(sns.regplot, lowess=True, scatter_kws={'alpha':.5}, line_kws={'colo
r': 'tomato'})
```

```
<Figure size 720x432 with 0 Axes>
```