Introduction to Data Analysis

Table of Content

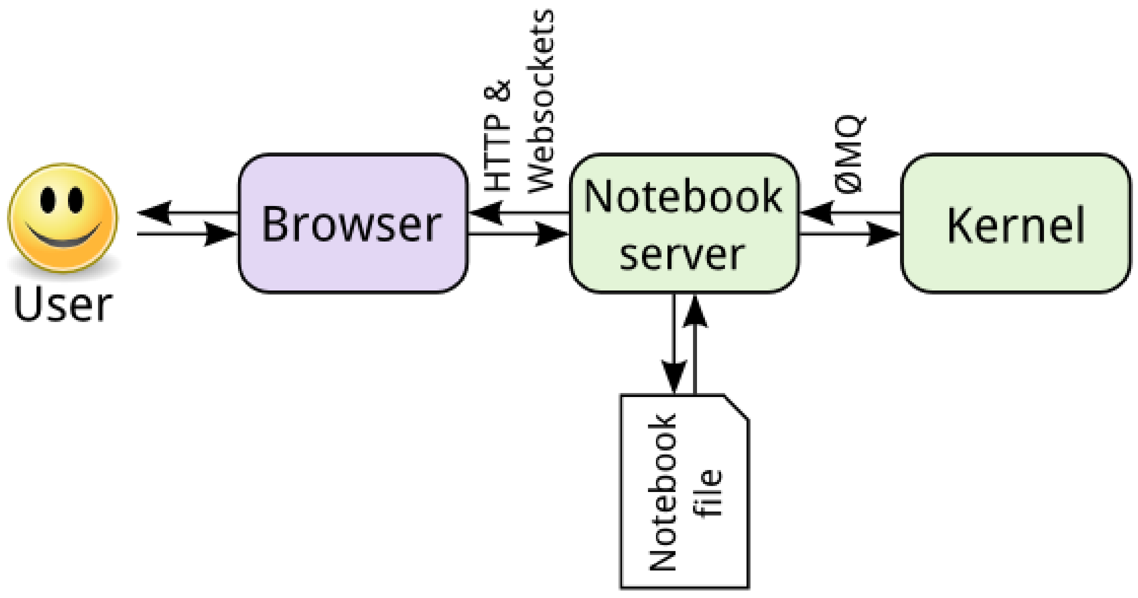
💡 **Moving Average**
Moving averages are used to smooth out data to make it easier to observe long term trends and not get lost in daily fluctuations.

💡 **Python Packages**
A package is a bunch of modules, where each module consists of a set of classes and function definitions. After installing a particular package, you can import and use the functions defined in that package.

💡 **Why do you need a Virtual Environment**
Each virtual environment remains isolated from other virtual environments, and the default system environment. Environments allow you to separate and isolate the packages you are using fro different projects.

```
#seach a package to install
conda search *SEARCH_TERM*
conda search *beautifulsoup*

#create environment
conda create -n env_name [python=X.X] [LIST_OF_PACKAGES]
conda create -n my_env python=3.7 numpy Keras

#entering/active an environment
conda activate my_env

#list the installed packages
conda list

#deactive an environment
conda deactivate

#saving and loading environment
conda env export
conda env export > environment.yaml

#create an environment
conda env create -f environment.yaml

#environments
conda env list
conda info --envs
conda list -n env_name
conda list -n env_name scipy
conda env remove -n env_name
```

How Notebooks Work
Jupyter notebooks grew out of the IPython project started by Fernando Perez. IPython is an interactive shell, similar to the normal Python shell but with great features like syntax highlighting and code completion.



Data Analysis Process

Step 1: Ask Questions

Great questions help you focus on relevant parts of your data and direct your analysis towards meaningful insights.

Step 2: Wrangle data

You get the data you need in a form you can work with in three steps: gather, assess, clean.

Step 3: Perform EDA (Exploratory Data Analysis)

Exploring involves finding patterns in your data, visualizing relationships in your data, and building intuition about what you're working with. After exploring, you can do things like remove outliers and create better features from your data, also known as feature engineering.

Step 4: Draw conclusions (or even make predictions)

This step is typically approached with machine learning or inferential statistics, which will focus on drawing conclusions with descriptive statistics.

Step 5: Communicate your results

You often need to justify and convey meaning in the insights you've found. Data visualization will always be very valuable.

```
#this returns a tuple of the simensions of the dataframe
df.shape

#this returns the datatypes of the columns
df.dtypes

#this displays a concise summary of the dataframe
#including the number of non-null values in each column
df.info()

#this returns the number of unique values in each column
df.nunique()

#this returns useful descriptive statistics for each column of data
df.dexcribe()

#view the index number and label for each column
for i,v in enumerate(df.columns):
    print(i,v)

#select all the columns from 'id' to the last mean column
df_means = df.loc[:,'id':'fractal_dimension_mean']

#repeat the step above using index numbers
df_means = df.iloc[:,:12]
```

```
#use np.r
np.r_[1:10, 15, 17, 50:100] #not including 10 and 100
df.iloc[:, np.r_[1:10, 15, 17, 50:100]]
```

💡 After assessing, you often need to fix problems in your data. Common problems include:

- Incorrect data type
- Missing data
- Duplicates
- Structural problems, such as different column names
- Mismatch number of records

```
#fill na
mean = df['view_duration'].mean()
df['view_duration'] = df['view_duration'].fillna(mean)

mean = df['column_name'].mean()
df['column_name'].fillna(mean, inplace = True)

#duplications
sum(df.duplicated())
df.drop_duplicates(inplace=True)

#timestamp
df['timestamp'] = pd.to_datetime(df['timestamp'])

#remove "_mean" from column names
new_labels = []
for col in df.columns:
    if '_mean' in col:
        new_labels.append(col[:-5])  # exclude last 6 characters
    else:
        new_labels.append(col)

#drop columns
df_18.drop(['Stnd', 'Stnd Description', 'Underhood ID', 'Comb CO2'],
              axis=1, inplace=True)

#rename columns
df_08.rename(columns={'Sales Area': 'Cert Region'}, inplace=True)

#replace spaces with underscores and lowercase labels
df_08.rename(columns=lambda x: x.strip().lower().replace(" ", "_"),
                inplace=True)

#confirm column are identical
df_08.columns == df_18.columns
# make sure they're all identical like this
(df_08.columns == df_18.columns).all()

#filter datasets for rows following California standards
df_08 = df_08.query('cert_region == "CA"')
df_18 = df_18.query('cert_region == “CA"')

#drop null
df_08.isnull().sum()
df_08.isnull().sum().any()
df_08.dropna(inplace=True)

#idxmax-index of the first occurrence of maximum value
idx = model_mpg.mpg_change.idxmax()
model_mpg.loc[idx]
```

💡 Histograms and scatterplots can help you determine which variables to use in your analysis.

💡 Normal distribution, also known as Gaussian distribution, is a probability distribution that is symmetric about the mean, showing that data near the mean are more frequent in occurrence than data far from the mean.

- A normal distribution is the proper term for a probability bell curve.
- In a normal distribution that mean is zero and the standard deviation is 1. It has zero skew and a kurtosis of 3.

- Normal distributions are symmetrical, but not all symmetrical distributions are normal.
- In reality, most pricing distributions are not perfectly normal.

💡 Here are four types of merges in pandas.

1. Inner Join - Use intersection of keys from both frames.
2. Outer Join - Use union of keys from both frames.
3. Left Join - Use keys from left frame only.
4. Right Join - Use keys from right frame only.

💡 IPython is actually what provides the interactive Python kernel we use in Jypter notebook. And in fact, we can use IPython outside of Jupyter notebook with its command line interface in our terminal.