Date:

→ PANDAS

correct syntax

→ create Pandas series from python list

↳ pd.Series(mylist)

→ First value of a Pandas series

↳ myseries[0]

→ to add lables "x", "y", "z" to pandas series

↳ pd.series(mylist, index=["x", "y", "z"])

→ to create Pandas dataframe

↳ pd.DataFrame(data)

→ to ~~create~~ return first row in a Pandas DataFrame

↳ df[0]

→ First row and second row in a Pandas Dataframe

↳ df.loc[[0, 1]]

retrieve syntax

→ return entire Dataframe

↳ df.to_string()

→ load JSON files into a Dataframe

↳ read_json()

→ Load python Dictionary called "data" into a Pandas Dataframe

↳ pd.Dataframe(data)

→ Pandas method for returning last rows

↳ tail()

→ removing rows that contain empty cells

↳ dropna()

→ fill empty cells with new values

↳ fillna()

→ to change original Dataframe instead of returning a new one.

↳ dropna(inplace=True)

Right syntax for

→ remove duplicates from a Pandas Dataframe.

↳ df.drop_duplicates()

→ to discover if a row is a duplicat

↳ df.duplicated()

→ To find relationships between column in a dataframe.

↳ df.corr()

→ to plot

↳ df.plot().

→ what is a dataframe?

↳ 2D, tabular data structur

more syntax:

df['columnName']: select a column

df.iloc[row-index, column-index]

↳ select specific row
by position.

df.loc[row-index, column-index] → by label

↳ Label : name or identifier of a row or column.

Position : numerical index or location of a row or columns.

df["NewColumn"] = values ; Add a column

df.drop('columnName', axis = 1)
↳ drop a column.

df.describe()
↳ summary statistics.

df.mean(), df.sum(), df.min().

Filtering
↳ df[df['Age'] > 30] :
filter rows based on a condition.

→ Pandas can clean messy data sets, and make them readable and relevant

→ Pandas Series.

↳ column in a table.

↳ 1D array holding data of any table

↳ Lables.

↳ If nothing is specified, the values are labeled with their index number.

↳ create Labels.

↳ code

↳ using Key/value objects as series

↳ Like dictionary.

↳ The keys of the dictionary become the labels.

Date :

→ Dataframes.

→ CSV : comma seperated values.

→ print(df.to_string())
  → to print entire data frame.

→ man_rows.
  → pd.options.display.man-rows
    → to check system's maximum rows. and also reset it.

→ JSON
  → Big datasets are often stored as JSON.
  → JSONs are Python dictionaries.

→ cleaning data in Pandas

→ df.dropna()

if you use
df.dropna(inplace=True)
→ changes the original dataframe

→ df.fillna

→ Replace values with some values.

→ fillna()

→ symmetrical distribution in histogram

→ mean

→ skewed

→ median

→ categorical

→ mode

Numpy

→ Library used for working with arrays

→ Provides an array object that is 50 x faster than traditional python lists.

→ ndarray

→ They are faster because data stored in one continuous place in memory.

→ slicing arrays.

→ arr = [1, 2, 3, 4]

→ arr[2:4] => [3,4]

→ arr[2:] => [3,4]

→ arr[:2] => [1,2]

→ arr[-2:-1] => [3]

→ arr[0:3:2] => [1,3]

→ arr[::2] => from 0 to east with step 2.

→ arr[1, 1:4]
→ starting from index 1 till 3rd index

→ arr = [[1, 2, 3, 4], [2, 3, 4, 5]]

arr[0:2, 3] -> [4, 5]

arr[0:2, 3] -> [4]

→ Data Types in Numpy.

→ i [Integer] : [-∞, ∞]
→ only integers

→ b [boolean] : [True, False]

→ u [unsigned Integer]
→ only positive whole numbers.

→ f [Float]
→ Decimal

→ c [complex]
→ 3 + 4j

→ m [Timedelta]
→ 5 day, 3 hours.

Interview Questions

→ Define LSTM

Ans LSTM is a type of RNN designed to handle the vanishing gradient problem, making it well-suited for tasks involving sequential data, such as time series analysis or NLP.
LSTMs can capture long-term dependencies in sequence, unlike traditional RNNs, which struggle to maintain information over longer time spans.

→ structure of LSTM

An LSTM consists of a series of gates that regulate the flow of information:

→ Forget Gate: Decides what information to discard from the cell state.

→ Input Gate: controls which new information is added to the cell state.

→ Cell State: carries the long-term memory, allowing information to persist over time.