Pandas - DA

Tuesday 18 March 2025 09:07

Data Analytics :

df = pd.read_csv('df')

- Data Exploration

df.shape / columns
df.describe()
df.info
} General

df.isna().values.any()
df.duplicated().values.any()
} Missing / Duplicates

df.isna().sum()
df.duplicated().sum()
→ duplicates = df[df.duplicated()]
to see duplicates/missing

Handle ⇒

Droping

clean = df.dropna() / df.dropna(inplace=True)
Clean = df.dropna(subset=['column_1', 'column_2'])
↑ droping in specific columns
clean = df.drop_duplicates() / df.drop_duplicates(inplace=True)
Clean2 = df.drop_duplicates(subset=['column_1', 'column_2'])
df = df.drop(columns='...', axis=1)

Filling Nan/ values → Median, Mean, Mode

df.fillna(0, inplace=True) / df['column'] = df['column'].fillna(0)
↑ to be replaced with
df['numeric_col'] = df['numeric_col'].fillna(df['numeric_col'].mean())
Median()
mode()[0] ← categorical data

Forward Fill / Backward Fill ← replacing with Next/Previous value

df.fillna(method='ffill', inplace=True)
'bfill'

Predicting

imputer = KNNImputer(n_neighbors=3)
df_imputed = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)

Journey

Loc & iloc

Name ↓ int position

df.loc['row1':'row3']

filtering = df.loc[df['column'] > threshold]

df.iloc[0,0] - First row, First column
df.iloc[0:3] - First 3 rows
df.iloc[:,1] - Second column
df.iloc[1:4, 0:2] - Specific rows, specific columns

Sorting & Grouping — Summarizing

df.sort_values("column", ascending=True)
df.value_counts() ← Grouping and counting only unique data
df.group_by('group')
df.group_by('group').size() ← Counting total + NaN
.count() ← Counts all existing values

agg()

Concerting data

df['date']=pd.to_datetime(df['date']) → df['year']=df['date'].dt.year/month/day
df['num'] = pd.to_numeric(df['num']) → Handling error : My data [1, 2, three, 4]
↑ Nan
df['num'], errors='coerce')
df['category'] = df['category'].astype('category')
df['Boolean'] = df['Boolean'].map({'true': True, 'false': False})
df['price'] = df['price'].replace('[\$]', '', regex=True).astype(float)

Querying

df.query("'column_1' > 2") - From this column_1 → show only values bigger than 2

ex_values = 5
df.query("'column_1' > @ex_values")

OR df = df[df['column_1'] < 5]

Outliners

- Take a look with Box plot

sns.boxplot(x=df['values'])
plt.title("Largest outliners")
plt.show()

- Take a look with histogram

df['values'].hist(bins=30)
plt.title(...)
plt.show()

Handling with Stats !

Q1 = df['values'].quantile(0.25)
Q3 = df['values'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
Outliners = df[(df['values'] < lower_bound | df['values'] > upper_bound)]
↳ Remove or Keep ← called Cap or Winsorize

Feature Engineering

- Creating new columns

df['new'] = df['old'] / df['old_2']

- Feature Extraction

df['year'] = df['date'].dt.year

- Feature transformation

Not for now