Pre-Processing in Structured Data Lab Session- I(b)

TIET, PATIALA

Pandas Introduction

Import pandas as pd

S.No	Feature	Syntax & Examples
1.	Creating Data Frame	df =pd.DataFrame()
2.	Adding Columns	df[‘Name’]=[‘abc’,’xyz’] df[‘age’]=[38,25]
3.	Loading a Data Frame	df=pd.read_csv(url/path) df=pd.read_csv(‘C:/Users/jasme/Desktop/titanic.csv’)
4.	Navigating Data Frame	df.iloc[row number/slice] df.iloc[4], df.iloc[1:4], df.iloc[:], df.iloc[1:4, 5:8]
5.	Conditional Row Selection	df[condition] df[df[‘Sex’]==‘female’] or df[(df[‘Sex’]==‘female’) &(df[‘Age’]>=‘65’)]

Pandas Introduction Contd….

Import pandas as pd

S.No	Feature	Syntax & Examples
6.	Replacing Values	df.replace(old_value,new_value) df.replace("female","Woman") df['Sex'].replace(["female","male"],["woman","man"])
7.	Renaming Columns	df.rename(columns={'Pclass':'Pessanger_Class'})
8.	Mathematical Functions	print(df['Age'].max()), print(df['Age'].min()) print(df['Age'].sum()), print(df['Age'].mean()) print(df['Age'].count())
9.	Unique Values	print(df['Sex'].unique()) print(df['Sex'].nunique()) print(df['Sex'].value_counts())
10.	Deleting Columns	df.drop(['Age'],axis=1) df.drop(df.columns[1],axis=1)

Pandas Introduction Contd….

Import pandas as pd

S.No	Feature	Syntax & Examples
11.	Deleting rows/duplicate rows	df[df['Sex']!='male’] or df.drop_duplicates()
12.	Grouping rows	df.groupby(‘Sex’), print(df.groupby('Age').sum())
13.	Looping over Column	[name.upper() for name in df['Name’]] Or for name in df['Name']: print(name.upper())
14.	Applying Functions Over all Elements of Column	df['Age'].apply(np.sqrt) df.groupby('Sex').apply(lambda x: x.count())

Data Cleaning- Missing Values

Feature	Syntax/ Example
1. Check Missing Values	print(pd.isna(df)) print(pd.isna(df['Pclass’]))
2. Deleting Missing Values Rows	df.dropna()
3. Replacing Missing value with scalar value	df['Age'].fillna(0) or df.fillna(0)
4 Fill NA forward and backward	df.fillna(method='pad’) df.fillna(method='bfill')
5. Central Tendency Imputation (mean, median, most_frequent, constant)	from sklearn.impute import SimpleImputer mean_imputer=SimpleImputer(missing_values=np.nan,strategy="mean") mean_imputer.fit_transform((df['Age'].values).reshape(1,-1))
6. Nearest Neighbor Imputer	from fancyimpute import KNN X_filled_knn = KNN(k=1).fit_transform((df['Age'].values).reshape(1,-1))

Data Cleaning- Noisy Data

Binning:

```
import pandas as pd
df=pd.read_csv('C:/Users/jasme/Desktop/titanic.csv')
import numpy as np
bins=np.linspace(min(df['Age']),max(df['Age']),4)
group_names=['child','young','old']
df['Age']=pd.cut(df['Age'],bins,group_names,include_lowest=True)
```

Data Cleaning- Handling Outliers

S.No	Feature	Syntax / Example
1.	Detecting Outliers (Box Plots, Scatter Plots)	import seaborn as sns sns.boxplot(x=df['Age'])
2.	Removing Outlier (using z score)	from scipy import stats import numpy as np df=df.fillna(method='pad') z = np.abs(stats.zscore(df['Age'])) print(np.where(z > 2.2))
3.	Removing Outlier (using z score)	Q1 = df.quantile(0.25) Q3 = df.quantile(0.75) IQR = Q3 - Q1 print(df < (Q1 - 1.5 * IQR)) |(df> (Q3 + 1.5 * IQR))

Data Transformation- Scaling

S.No	Feature	Synatx Example
1.	MinMax Scaler	from sklearn.preprocessing import MinMaxScaler scaler = MinMaxScaler() df['Age']=scaler.fit_transform(df['Age'].values.reshape(-1, 1)).flatten()
2	Standard Scaler (z-score)	from sklearn.preprocessing import StandardScaler scaler = StandardScaler() df['Age']=scaler.fit_transform(df['Age'].values.reshape(-1, 1)).flatten()
3	Robust Scaler (IQR)	from sklearn.preprocessing import RobustScaler scaler = RobustScaler() df['Age']=scaler.fit_transform(df['Age'].values.reshape(-1, 1)).flatten()
4	MaxAbs Scaler	from sklearn.preprocessing import MaxAbsScaler scaler =MaxAbsScaler() df['Age']=scaler.fit_transform(df['Age'].values.reshape(-1, 1)).flatten()

Data Transformation- Encoding

S.No	Feature	Synatx Example
1.	Label Encoder	from sklearn import preprocessing le = preprocessing.LabelEncoder() le.fit([1, 2, 2, 6])
2	One Hot Encoder	from sklearn.preprocessing import OneHotEncoder onehotencoder = OneHotEncoder() onehotencoder.fit_transform(data)