23CSE301 Machine Learning

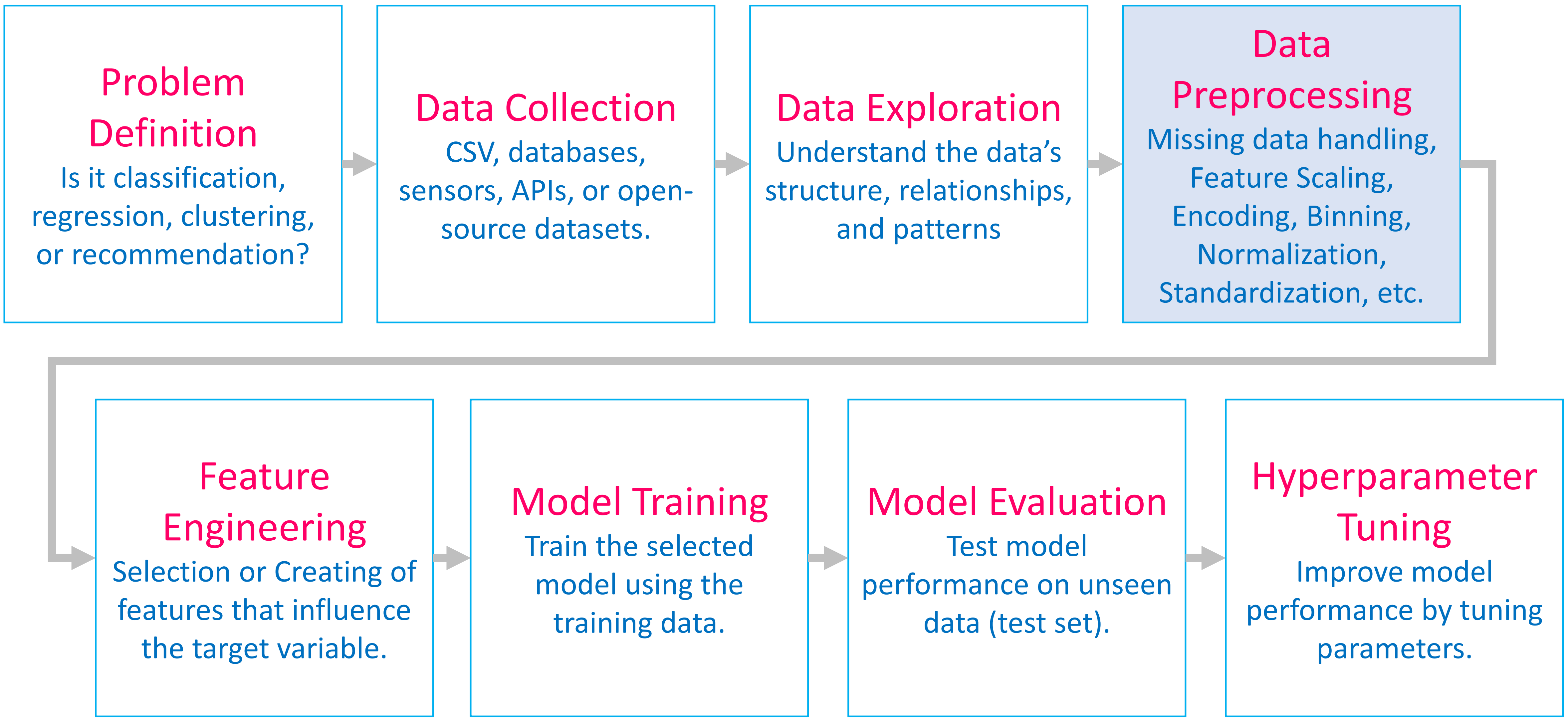
V Sem. CSE B

Practical – Week 4

Course Instructor: Dr. M. Anbazhagan

Pract. #	Experiment Title
P1-P3	• Introduction: Python, Pandas (scikit learn and other libraries) • Pre-processing: Dataset selection, Exploratory Data analysis and Feature engineering; Introduction to Colab/Jupyter Notebook, Pandas(Data Frames); Data Selection (iloc, loc); Sorting, Grouping merge, join, concat; Crosstab; Missing data treatment(fillna, dropna), Converting categorical values, Visualization(Line chart, Bar Chart, Pie chart, Scatter plot, Box plot); Distributions; Summary statistics.
Lab 1 Evaluation (P1 to P3)	
P4	Dimensionality Reduction Technique: PCA
P5	Feature Selection
P6	Regression Algorithms: Linear Regression
P7	Regression Algorithms: Logistic Regression
P8	Classification Algorithms: Decision Tree Classifier
	Classification Algorithms: K-Nearest Neighbor Classifier
Lab 2 Mid-Term exam (P1 to P8)	
P9	Classification Algorithms: Random Forest Classifier, ensemble learning.
P10	Classification Algorithms: Support Vector Machines
P11	Classification Algorithms: Perceptron
P12	Clustering: 1. K-Means Clustering 2. Agglomerative Clustering
Lab 3 Evaluation (P1 to P12)	

End-to-End Machine Learning Pipeline



Handling Missing Data

Handling missing data is a crucial step in data preprocessing. Here are six commonly used methods:

Simple Imputer
Replaces missing values with a constant or a statistical value like the mean, median, or mode of the column

Forward Fill (Pad)
Propagates the last valid observation forward to fill gaps

Backward Fill
Uses the next valid observation to fill missing values

KNN Imputer
Fills in missing values using the average of the nearest neighbors' values

Model-Based Imputation
Trains a predictive model to estimate missing values based on other features

Deletion
Removes rows (or columns) with missing values

Handling Outliers

Outliers are data points that differ significantly from the rest of a dataset. They stand out because they're either much higher or lower than the typical values, and they can reveal interesting insights or cause misleading results if not handled properly.

Outlier Detection Methods

1. Visualization-Based
   - Boxplot
   - Histogram
   - Scatterplot
2. Statistical Method
   - IQR Method
   - Z-Score Method

Outlier Handling Methods

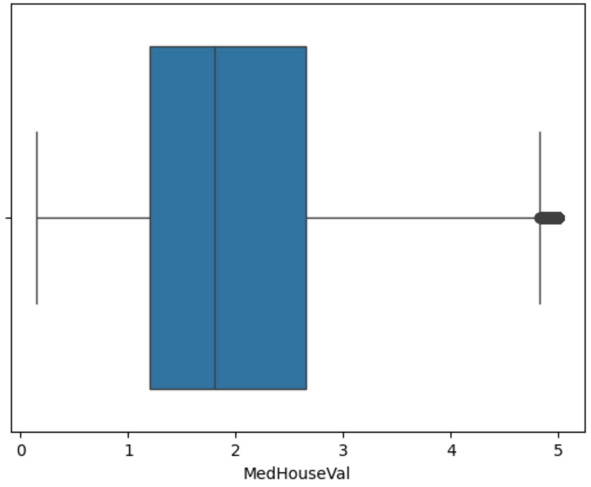
1. Removal
2. Capping / Winsorizing
3. Transformation
4. Imputation
5. Use Robust Algorithms

Outlier Detection

Boxplot

Outliers are points outside the whiskers

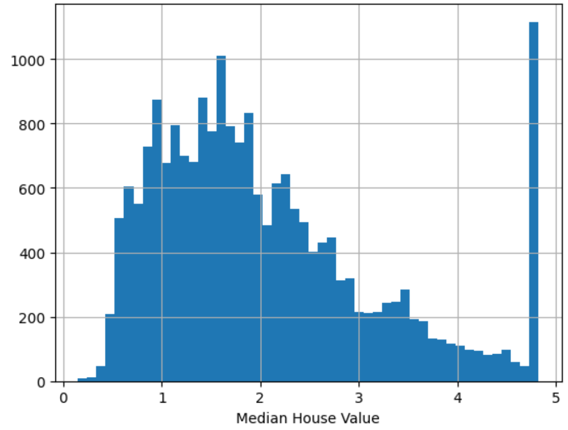
```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
data = fetch_california_housing(as_frame=True)
df = data.frame
sns.boxplot(x=df[‘MedHouseVal’])
plt.show()
```



Histogram

Look for extreme left/right tail values

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
data = fetch_california_housing(as_frame=True)
df = data.frame
df['MedHouseVal'].hist(bins=50)
plt.xlabel('Median House Value')
plt.show
```



Outlier Detection

Z-Score

Values greater than |3| standard deviations from the mean are flagged

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

z_scores = np.abs(stats.zscore(df['MedHouseVal']))
outliers_z = df[z_scores > 3]
print(outliers_z.shape)
```

IQR

Outliers lie below Q1 - 1.5×IQR or above Q3 + 1.5×IQR.pythonCopyEdit

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
data = fetch_california_housing(as_frame=True)
df = data.frame

Q1 = df['MedHouseVal'].quantile(0.25)
Q3 = df['MedHouseVal'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
outliers_iqr = df[(df['MedHouseVal'] < lower_bound) |
(df['MedHouseVal'] > upper_bound)]
print(outliers_iqr.shape)
```

Handling Outliers

Remove Outliers

Drop outlier rows entirely

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

df_removed = df[(df['MedHouseVal'] >= lower_bound) & (df['MedHouseVal'] <= upper_bound)]
print(df_removed.shape)
```

Capping / Winsorizing

Replace extreme values with the nearest acceptable boundary

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
data = fetch_california_housing(as_frame=True)
df = data.frame

df_capped = df.copy()
df_capped['MedHouseVal'] = np.where(df_capped['MedHouseVal'] > upper_bound, upper_bound,

np.where(df_capped['MedHouseVal'] < lower_bound, lower_bound, df_capped['MedHouseVal']))
```

Handling Outliers

Log Transformation

Apply log, square root, or Box-Cox transformations to reduce skewness and outlier impact

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

df_log = df.copy()
df_log['MedHouseVal'] =
np.log(df_log['MedHouseVal'] + 1)
```

Imputation

Replace outliers with mean/median/mode

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
import numpy as np
data = fetch_california_housing(as_frame=True)
df = data.frame

df_imp = df.copy()
median_value = df['MedHouseVal'].median()
df_imp.loc[(df_imp['MedHouseVal'] > upper_bound) |
(df_imp['MedHouseVal'] < lower_bound),
'MedHouseVal'] = median_value
```

Exercise 1 - Week 4

- Dataset: Heart Failure Clinical Records Dataset
  - You are provided with the Heart Failure Clinical Records Dataset, which contains clinical and demographic information about heart failure patients. Load this dataset into a Pandas DataFrame using Jupyter Notebook and begin with dataset selection, inspecting its structure, and performing exploratory data analysis. Use Pandas operations like iloc, loc, sorting, grouping, merging, joining, and crosstabs to understand relationships between different variables such as age, ejection fraction, and survival status. Handle missing data using appropriate techniques like fillna() or dropna() and perform necessary feature engineering tasks such as converting categorical values into numeric form. Visualize key variables and relationships using line charts, bar charts, pie charts, scatter plots, box plots, and distribution plots. Conclude by generating summary statistics and interpreting the distributions of numerical variables to uncover trends and patterns that could be important for predicting patient outcomes.

Exercise 2 - Week 4

- Dataset: Credit Scoring Dataset
  - Using the Credit Scoring Dataset, perform a detailed data exploration and preprocessing workflow. Begin by importing the dataset into a Pandas DataFrame and inspecting its structure, dimensions, and basic statistics. Proceed with exploratory data analysis by grouping and sorting records based on credit status, employment length, and income level, and use Pandas methods like iloc, loc, and crosstab for targeted data inspection. Give special attention to detecting outliers in numerical columns such as income, loan_amount, and years_employed using boxplots, Z-Score, and IQR methods. Handle these outliers either by capping, removing, or transforming them, and perform missing data treatment where necessary. Convert any categorical columns into numeric form if needed, and visualize your cleaned data using appropriate charts such as bar plots, pie charts, scatter plots, and distribution plots. Wrap up by generating summary statistics and distribution insights for key features and reflect on how preprocessing choices affected data interpretation.