Week -2

1 Get the data

Downloading - data -

```
import os
import tarfile
from six.moves import urllib

DOWNLOAD_ROOT = "http://raw.githubusercontent.com/
                 ageron/handson-ml2/master/"
HOUSING_PATH = os.path.join("data", "01")
HOUSING_URL = download_root + "datasets/
              housing/housing.tgz"

def fetch_housing_data(housing_url=
    Housing_url, Housing_path=
        Housing_path)

os.makdirs(name = housing_path,
           exist_ok = True)

tgz_path = os.path.join(housing_path,
           housing.tgz")

urllib.request.req(url= housing_path,
                   "housing.tgz)

housing_tgz.close().
```

→ Create test set

```
import numpy as np

def split_train_test(data, test_ratio=0.2):
    shuffled_indices = np.random.permutation(len(data))
    test_set_size = int(len(data) * test_ratio)
    test_indices = shuffled_indices[:test_set_size]
    train_indices = shuffled_indices[test_set_size:]
    return data.iloc[train_indices], data.iloc[test_indices]
```

⇒ Discover & visualize Data to gain insights

```
strat_train_set.shape, strat_test_set.shape

strat_test_set.reset_index().to_feather('data/02/strat_test_set.f')

housing = strat_train_set.copy();
housing.shape
```

→ Prepare Data for Machine Learning Model

```
housing = strat_train_set.drop("median_house_value", axis=1)
housing_labels = strat_train_set["median_house_value"].copy()
```

housing.shape, housing_label.shape

→ Select & Train Model

```
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression()
lin_reg.fit(X = housing_prepared,
            y = housing_labels).
```

04/11/24

Week -3

Multiple Linear Regression

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import
train_test_split
from sklearn.linear_model import
LinearRegression
```

⇒ Import data

```
df_start = pd.read_csv('/content/50
                        _Startups.csv')
df_start.head()
```

⇒ Describe

```
df_start.describe()
```

⇒ Distribution

```
plt.title('Profit Distribution Plot')
sns.distplot(df_start['Profit'])
plt.show()
```

Relationship b/w Profit & R&D Spend

```
plt.scatter(df_start['R&D Spend'],
            df_start['Profit'], color='lightcoral')
plt.title('Profit vs R&D Spend')
plt.xlabel('R&D Spend')
plt.ylabel('Profit')
plt.box(False)
plt.show()
```

Train Model

```
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Predict Results

```
y_pred = regressor.predict(X_test)
```

Compare predictions

```
np.set_printoptions(precision=2)
result = np.concatenate((y_pred.
reshape(len(y_pred),1), y_test.
reshape(len(y_test),1)),1)
result.
```

4/4/24

Lab - 5

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
df = pd.read_csv("insurance_data.csv")
df.head()
plt.scatter(df.age, df.bought_insurance,
    marker='+', color='r')
from sklearn.model_selection import
train_test_split
X_train, X_test, y_train, y_test =
train_test_split(df[['age']], df.bought_
insurance, train_size=0.8)
from sklearn.linear_model
import LogisticRegression
model = LogisticRegression()
model.fit(X_train, y_train)
y_predicted = model.predict(X_test)
model.predict_proba(X_test)
```

model.score_proba(X_test)

y_predicted

X_test

model.coef_

model.intercept_

```
import math
def sigmoid(x):
    return 1/(1+math.exp(-x))
def prediction_function(age):
    z= 0.042 * age -1.53
    y = sigmoid(z)
    return y
```

age

prediction_function(age)

age = 43

prediction_function(age)

0.5185652990777705

25/4/24

Lab - 4

Program - 4

Linear Regression

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
df = start.head()
df.describe()
```

Distribution

```
plt.title('profit Distribution plot')
sns.displt(df-start['profit'])
plt.show()
```

split Data

```
Y = df-start[:, :-1].values
Y = df-start[:, -1].value
```

split into train / test data

```
X_trans, X_test, Y train, Y_test
    train_test_split(x, y, test_size=0.2
                     random_stame=0)
```

~~Lab-6-7~~

Program 6-7

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import
test.train
from sklearn.preprocessing import label
encoder
from sklearn.metrics import accuracy_score
from sklearn.svm import SVC
data = pd.read_csv('iris.csv')
print(data.head())
plt.figure(figure=(10,6)).
plt.title('Scatter plot of Iris Data')
plt.show()
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size
                 0.2
                 random =52)
```

3/6/24

Program-8

ANN

```
import numpy as np
x = np.array([[2,9],[1,5],[3,6]], dtype=float)
y = np.array([[92],[86],[89]], dtype=float)
x = x/np.amax(x-axis=0)
y = y/100
epoch = 5000
lr = 0.1
inputlayer_neurons = 2
hiddenlayer_neurons = 3
output_neurons = 1
wh = np.random.uniform(size=(inputlayer_neurons, hiddenlayer_neurons))
bh = np.random.uniform(size=(1, hiddenlayer_neurons))
wout = np.random.uniform(size=(hiddenlayer_neurons, outputneurons))
bout = np.random.uniform(size=(1, outputneurons))
```

print ('Input \n" + str (x))
print ("Actual Output \n", str (y))
print ("predicted Output \n", output)

Input:

[0.666667 . 1.]
[0.3 3333 0.55 556]
[1. 0.666777]

Program -99

Random Forest

```
import pandas as pd

from sklearn.model_selection import
    train_test_split

from sklearn.ensemble import
    RandomForestClassifier

from sklearn.dataset import load_
    iris

from sklearn.model_selection
    import train_test_split.

iris_data = pd.read_csv('Iris.csv')
iris = data.head()

X_train, X_test, Y_train, Y_test
    = train_test_split(X, Y, test_
                size = 0.4,
                random_state = 42)

rf_classifier = RandomForestClassifier
                (n_estimators =
                100, random_state 42)
rf_classifier.fit(X_train, Y_train)
```

30/5/24

Program - 9b

```
iris.data = pd.read_csv('Iris.csv')
iris.data.drop("species", axis=1)
iris.data.head()
x = iris_data.drip('species', axis=1)
y = iris.data.info()
X_train, X_test, y_train, y_test,
train_test_split(x, y, test_size=0.4
random_state=42)
mylogregonodd = logisticRegression()
adaclc = AdaBoatClassifier(
                X_train, y_train.
accuracy = accuracy_score(y_test,
                y_pred).
```

Program 10-a

(Breast Cancer - PCA)

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
% matplotlib inline

from sklearn.datasets import
load_breast_cancer.

cancer = load_breast_cancer()

cancer.keys()

print(cancer['DESCR'])

df = pd.DataFrame(cancer['data'],
columns = cancer['feature_names'])

df.head()

cancer['target_names']

scaler = StandardScaler()

scaler.fit(df)

x_pca = pca.transform(scaled_data)

x_pca.shape
```

Program 11

K means - Clustering of Iris

```
import os
import sys
import tempfile import NamedTemporaryFile
import zipfile import ZipFile
import tarfile
import shutil

CHUNK_SIZE = 40960
DATA_SOURCE_MAPPING = 'iris-flower-dataset'

try:
    os.symlink(KAGGLE_INPUT_PATH, os.path.
        join("..", 'input'), target_is_
        directory = TRUE)

except FileExistsError:
    pass

try:
    os.symlink(KAGGLE_Working Path,
        os.path.join("..", 'working'),
        target_is_directory = True

except FileExistsError:
    pass.
```

30/6/24

INDEX

NAME: SNEHAL BANDI STD: SEC: ROLL NO:

Sl No	Date	Title	Page No.	Teacher's Sign / Remarks
1.	21-3-24	Importing & Exporting Data wo using library		4/4/24
2.	28-3-24	End to end ML Project		4/4/24
3.	4/4-24	Linear Regression		9/5/24
4.	18-4-24	Decision tree		
5.	20-4-24	Logistic Regression		9/5/24
6	9-5-24	Build KNN Classification		
7.	9-5-24	Build Support vector machine model for a given dataset		30/5
8.	23-5-24	Artificial Network Model		30/5
9.	23-5-24	Random Forest, Adaboost		30/5/24
10.	30-5-24	Breast Cancer PCA		
11	30-5-24	KMeans Clustering.		