Index

S. No.	Name of the Experiment	Page No.	Date of Experiment	Date of Submission	Remark
1.	Implement FIND-S Algorithm	1-2			
2.	Implement K-Nearest Neighbors	3			
3.	Implement Linear Regression	4			
4.	Implement Ridge	5			
5.	Implement Lasso	6			
6.	Implement Logistic Regression	7			
7.	To implement LinearSVC	8			
8.	To implement DecisionTree	9			

Date _____

Expt. No. 01 EXPERIMENT - 01 Page No. 01

AIM: Implement FIND-S Algorithm

Code:

```
import pandas as pd
import numpy as np

print("Find s Algorithm")
data = pd.read_csv("finds.csv")
print("Dataset = ")
print(data)

d = np.array(data)[:, :-1]
print("\n\nThe attributes are: ", d)

target = np.array(data)[:, -1]
print("\n\nThe target is: ", target)

def train(c,t):
    for i, val in enumerate(t):
        if val == "Yes":
            specific_hypothesis = c[i].copy()
            break

    for i, val in enumerate(c):
        if t[i] == "Yes":
            for x in range(len(specific_hypothesis)):
                if val[x] != specific_hypothesis[x]:
                    specific_hypothesis[x] = '?'
```

Teacher's Signature _____

```
        else:
            pass
    return Specific_hypothesis

print("\n\n The final hypothesis is:", train(d, target))
```

Date

Expt. No. 02 EXPERIMENT- 02 Page No. 03

AIM: Implement K-Nearest Neighbors

Code:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_breast_cancer

cancer = load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(cancer.data,
                                   cancer.target, random_state = 0)

model = KNeighborsClassifier(n_neighbors = 5)

print("Prediction = {}".format(model.predict(X_test)))

print("Train score = {}".format(model.score(X_train, y_train)))

print("Test score = {}".format(model.score(X_test, y_test)))
```

Output:

Prediction = [0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 1 0
1 1 1 1 0 1 0 1 0 1 0 1 - - - 1 1 1 0]

Train score = 0.9413145539

Test score = 0.9370629370

Teacher's Signature

Date _____

Expt. No. 03 EXPERIMENT - 03 Page No. 04

AIM: Implement Linear Regression

Code:

```
! pip install mglearn -q
import mglearn
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

X, y = mglearn.datasets.load_extended_boston()
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                   random_state = 0)
model = LinearRegression()
model.fit(X_train, y_train)

print("Prediction = {}".format(model.predict(X_test)))
print("Train score = {}".format(model.score(X_train, y_train)))
print("Test score = {}".format(model.score(X_test, y_test)))
```

Output:

```
Prediction = [23.6529   26.73756   29.6107   10.15490293
              19.6956   22.04569   20.8219   11.62647
                  :         :         :         :
              19.28660344   36.92350037]
Train score = 0.9520
Test score = 0.6074
```

Teacher's Signature _____

Date _____

Expt. No. 04 EXPERIMENT - 04 Page No. 05

AIM: Implement Ridge

Code:

```
!pip install mglearn -q
import mglearn
from sklearn.linear_model import Ridge
from sklearn.model_selection import train_test_split

X, y = mglearn.datasets.load_extended_boston()
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                   random_state = 0)
model = Ridge(alpha = 0.10)

model.fit(X_train, y_train)
print("Prediction = {}".format(model.predict(X_test)))
print("Train score = {}".format(model.score(X_train, y_train)))
print("Test score = {}".format(model.score(X_test, y_test)))
```

Output:

```
Prediction = [24.7489   26.0155   27.0821   12.7116
              19.7578   19.13628  22.3630   21.54557
                 .         .         .         .
              35.76995172]
Train score = 0.928227
Test score = 0.772206
```

Teacher's Signature _____

Date _____

Expt. No. 05 EXPERIMENT – 05 Page No. 06

AIM: Implement lasso

Code:

```
!pip install mglearn -q
import mglearn
from sklearn.linear_model import Lasso
from sklearn.model_selection import train_test_split

X, y = mglearn.datasets.load_extended_boston()
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                   random_state = 0)
model = Lasso(alpha = 0.01, max_iter = 10000)
model.fit(X_train, y_train)
print("Prediction = {}", format(model.predict(X_test)))
print("Train score = {}", format(model.score(X_train, y_train)))
print("Test score = {}", format(model.score(X_test, y_test)))
```

Output:

```
Prediction = [24.2233  24.7780  26.7108  13.3084
              20.6375  19.4827  21.9224  21.3672
                  !        !        !        !
              35.6097]
Train score = 0.8962
Test score = 0.7656
```

Teacher's Signature _____

Date

Expt. No. 06 EXPERIMENT- 06 Page No. 07

AIM: Implement Logistic Regression

Code:

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_breast_cancer

cancer = load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(cancer.
                    data, cancer.target, random_state = 0)
model = LogisticRegression()
model.fit(X_train, y_train)
print("Prediction = {}".format(model.predict(X_test)))
print("Train score = {}".format(model.score(X_train, y_train)))
print("Test score = {}".format(model.score(X_test, y_test)))
```

output:

```
Prediction = [0 1 1 1 1 1 1 1 0 1 0 1 0 0 0 1 1 -
- - - - - - - - - - - - - - - - - - - -
0 0 1 1 1 1 1 1 0 1 0 1 0 1 1 0 1 1 0 1 1 1 0]
```

Train score = 0.9577

Test score = 0.95804

Teacher's Signature

Date _____

Expt. No. 07 EXPERIMENT - 07 Page No. 08

AIM: To Implement Linear SVC

Code:

```
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_breast_cancer

cancer = load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(cancer.data,
                                   cancer.target, random_state = 0)
model = LinearSVC()
model.fit(X_train, y_train)
print("Prediction = {}".format(model.predict(X_test)))
print("Train score = {}".format(model.score(X_train, y_train)))
print("Test score = {}".format(model.score(X_test, y_test)))
```

Output:

Prediction = [0 1 1 1 1 1 1 1 1 0 1 0 1 0 0 1 1 1 1 0 1 0 1 ... 1 1 1 0]

Train score = 0.892018

Test score = 0.8881118

Teacher's Signature _____

Date _____

Expt. No. 08 — EXPERIMENT- 08 — Page No. 09

AIM: To implement DecisionTree

Code:

```
from SKlearn.tree import DecisionTreeClassifier
from SKlearn.model_selection import train_test_split
from SKlearn.datasets import load_breast_cancer

cancer = load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(cancer.
                    data, cancer.target, random_state = 0)
model = DecisionTreeClassifier(random_state = 0)
model.fit(X_train, y_train)

print("Prediction = {}".format(model.predict(X_test)))
print("Train score = {}".format(model.score(X_train, y_train)))
print("Test score = {}".format(model.score(X_test, y_test)))
```

Output:

```
Prediction = [0 1 1 1 1 1 1 0 1 1 0 0 0 1 0 0 0 0 1 1 0
              1 1 1 1 0 1 1 1 1 0]

Train score = 1.0
Test score = 0.881188
```