**MACHINE LEARNING**

ASSIGNMENT 2

REPORT

**TEAM MEMBERS**

R LOGESHWARAN 19S016

S SANTHOSH KUMAR 19S030

K R VISHNU KARTHICK 19S039

**Introduction:**

This is an Assignment where we have used a dataset and have applied algorithms like Gradient Descent, Feature spacing and the conventional classification model and we will be comparing the key outcomes of each algorithm and will conclude which is the best.

**About Dataset:**

This is a simple dataset on Social network advertisements which had made people buy products online. The dataset contains information on the buyer’s gender, age and salary and whether they have purchased products or not. This dataset contains about 400 rows of data. This dataset has been from Kaggle.

<https://www.kaggle.com/dragonheir/logistic-regression>

**Algorithms Used:**

1. Gradient Descent:

Gradient descent is an optimization algorithm that follows the negative gradient of an objective function in order to locate the minimum of the function.

It is a simple and effective technique that can be implemented with just a few lines of code. It also provides the basis for many extensions and modifications that can result in better performance. The algorithm also provides the basis for the widely used extension called stochastic gradient descent, used to train deep learning neural networks.

References:

<https://www.datatechnotes.com/2020/09/sgd-classification-example-with-sgdclassifier-in-python.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html>

1. LDA-Linear Discriminant Analysis:

Linear Discriminant Analysis (LDA) is a dimensionality reduction technique. As the name implies dimensionality reduction techniques reduce the number of dimensions (i.e. variables) in a dataset while retaining as much information as possible. Linear Discriminant Analysis, or LDA, uses the information from both features to create a new axis and projects the data on to the new axis in such a way as to minimizes the variance and maximizes the distance between the means of the two classes.

<https://towardsdatascience.com/linear-discriminant-analysis-in-python-76b8b17817c2>

1. Logistic Regression:

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability.We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the ‘Sigmoid function’ or also known as the ‘logistic function’ instead of a linear function.

<https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148>

**SOURCE CODE:**

1. **STOCHASTIC GRADIENT DESCENT:**

import pandas as pd

import numpy as np

data = pd.read\_csv("Social\_Network\_Ads.csv")

data.head()

data.info()

data**.**isnull()**.**sum()

data.shape

data.describe()

data.nunique()

data = data.drop(["User ID"],axis=1)

data.head()

data['Gender'] = pd.factorize(data.Gender)[0]

data.head()

N = ['Age','EstimatedSalary']

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

xscaled = scaler.fit\_transform(data[N])

data.loc[:, N] = xscaled

data.head()

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import SGDClassifier

y=data.pop('Purchased')

X=data

X.head()

y.head()

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y, test\_size = 0.2, random\_state = 10)

X\_train.shape

sgdc = SGDClassifier(max\_iter=10000, tol=0.01)

sgdc.fit(X\_train, y\_train)

score = sgdc.score(X\_train, y\_train)

print("Training Set Score: ", score)

from sklearn.metrics import confusion\_matrix

ypred = sgdc.predict(X\_test)

cm = confusion\_matrix(y\_test, ypred)

print(cm)

from sklearn.metrics import classification\_report

from sklearn.metrics import accuracy\_score

cr = accuracy\_score(y\_test, ypred)

print(cr)

cr = classification\_report(y\_test, ypred)

print(cr)

1. **LDA-Linear Discriminant Analysis:**

# In[1]:

import pandas as pd

# In[2]:

df = pd.read\_csv('Social\_Network\_Ads.csv')

df['Gender'] = pd.factorize(df.Gender)[0]

y=df.pop('Purchased')

X=df

# In[3]:

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

# In[4]:

x\_train,x\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state=42)

# In[5]:

model = LogisticRegression()

model.fit(x\_train,y\_train)

y\_pred = model.predict(x\_test)

print(accuracy\_score(y\_pred,y\_test))

# In[6]:

from sklearn.discriminant\_analysis import LinearDiscriminantAnalysis

# In[7]:

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y,test\_size=0.2,random\_state=42)

lda = LinearDiscriminantAnalysis(n\_components=1)

lda.fit(x\_train, y\_train)

# In[8]:

from sklearn.metrics import accuracy\_score

y\_pred = lda.predict(x\_test)

print(accuracy\_score(y\_test, y\_pred))

# In[10]:

from sklearn.metrics import classification\_report

cr = classification\_report(y\_test, y\_pred)

print(cr)

1. **LOGISTIC REGRESSION:**

# In[1]:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# In[2]:

data = pd.read\_csv('Social\_Network\_Ads.csv')

data.head()

# In[3]:

data.info()

# In[4]:

data.isnull().sum()

# In[5]:

data.describe()

# In[6]:

data['Gender'] = pd.factorize(data.Gender)[0]

# In[7]:

data = data.drop(['User ID'],axis =1)

# In[8]:

X=data.drop(['Purchased'],axis=1)

y=data.drop(['Gender','Age','EstimatedSalary'],axis=1)

# In[9]:

X.head()

# In[10]:

y.head()

# In[11]:

from sklearn.model\_selection import train\_test\_split

# In[12]:

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y, test\_size = 0.3, random\_state = 100)

X\_train.shape

# In[13]:

from sklearn.linear\_model import LogisticRegression

# In[14]:

log\_regression = LogisticRegression()

log\_regression.fit(X\_train,y\_train)

y\_pred = log\_regression.predict(X\_test)

ynew = log\_regression.predict(X)

# In[15]:

from sklearn import metrics

# In[16]:

print("Accuracy:",metrics.accuracy\_score(y\_test,y\_pred))

# In[17]:

log\_regression.coef\_

# In[18]:

from sklearn.metrics import classification\_report

# In[20]:

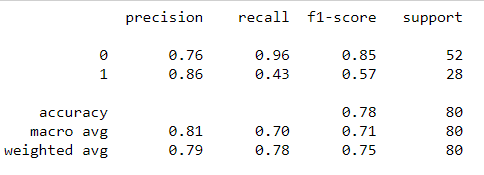
cr = classification\_report(y\_test, y\_pred)

print(cr)

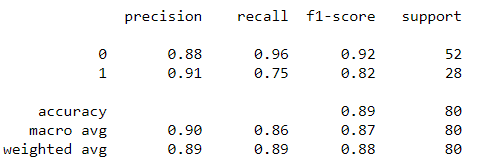
**COMPARISON:**

Classification Reports :

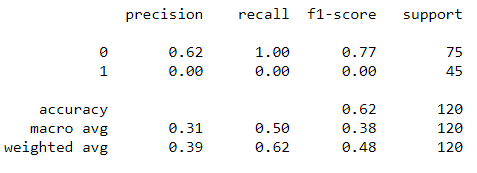
Gradient Descent:



LDA:



LOGISTIC REGRESSION:



| GRADIENT DESCENT | LDA ANALYSIS | LOGISTIC REGRESSION |
| --- | --- | --- |
| * This Algorithm gave us the one of the highest accuracies. The accuracy was found out to be in the range of 0.79 - 0.89. | * This algorithm gave us an accuracy of   0.8875. | * This basic classification algorithm rendered us with the lowest accuracy. The accuracy was about 0.625. |

**CONCLUSION:**

For this dataset, Stochastic Gradient descent and LDA produced similar results, we can arrive at the fact that both these algorithms are best for this dataset. But, From the most common results, LDA is said to be best for bivariate classification problems but it is not that strong compared to Logistic Regression in the cases of Multivariate classification problems. So we can conclude the fact that Gradient descent is more stable and reliable for multivariate classification problems.