Dicision Tree

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

df= pd.read_csv("/home/ignis/zoo11.csv")

df.head()

class_type_output = df["class_type"]

df = df.drop("class_type", axis=1).drop("animal_name",axis=1)

print(df)

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(df, class_type_output,
test_size=0.20)

from sklearn.tree import DecisionTreeClassifier

classifier = DecisionTreeClassifier()

classifier.fit(x_train, y_train)

y_prediction = classifier.predict(x_test)

y_prediction

from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

confusion_matrix(y_test,y_prediction)

print(classification_report(y_test, y_prediction))

print(accuracy_score(y_test, y_prediction))

predicted_class = list(y_prediction)

actual_class = list(y_test)

for i in range(len(predicted_class)):
    print("Predicted class =", predicted_class[i],"\tActual class =",actual_class[i])
```

SVC

```
from sklearn.svm import SVC
from sklearn import svm
import numpy as np
X=np.array([[3,4],[1,4],[2,3],[6,-1],[7,-1],[5,-3]])
y=np.array([-1,-1,-1,1,1,1])
l=SVC(C=1e5,kernel='linear')
l.fit(X,y)
print('w= ',l.coef_)
print('b= ',l.intercept_)
print('Indices of support vectors= ',l.support_)
print('Support vectors= ',l.support_vectors_)
print('No. of support vectors from each class= ',l.n_support_)
print('coefficient of support vectors in decision function= ',np.abs(l.dual_coef_))

import pandas as pd
data=pd.read_csv('/home/ignis/Downloads/glass.csv')
data.head()

x=data.drop('Type',axis=1)
y=data.Type
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
linear=svm.SVC(kernel='linear')
linear.fit(x_train,y_train)
print(linear.support_vectors_)
```

```
print(linear.n_support_)
y_pred=linear.predict(x_test)


from sklearn.metrics import accuracy_score
print(accuracy_score(y_test,y_pred))
from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test,y_pred))
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
model1=SVC(kernel='sigmoid')
model2=SVC(kernel='poly')
model3=SVC(kernel='rbf')
model1.fit(x_train,y_train)
model2.fit(x_train,y_train)
model3.fit(x_train,y_train)
y_pred1=model1.predict(x_test)
y_pred2=model2.predict(x_test)
y_pred3=model3.predict(x_test)
print("prediction by model1 ",accuracy_score(y_test,y_pred1))
print("prediction by model2",accuracy_score(y_test,y_pred2))
print("prediction by model3",accuracy_score(y_test,y_pred1))
```

PCA

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
df=pd.read_csv('/home/root1/pca.csv')
df.head()



X = df.drop(['species'],axis=1)
X_scaled = StandardScaler().fit_transform(X)
X_scaled[:5]
y=df['species']
/home/root1/Desktop
features = X_scaled.T
cov_matrix = np.cov(features)
cov_matrix[:5]
values, vectors = np.linalg.eig(cov_matrix)
values[:5]
vectors[:5]
explained_variances = []
for i in range(len(values)):
    explained_variances.append((values[i] / np.sum(values))*100)
print("variances of each feature",explained_variances)
plt.figure(figsize=(8,4))
plt.bar(range(4),explained_variances, alpha=0.6)
```

```
plt.ylabel('Percentage of explained variance')
plt.xlabel('Dimensions')
projected_1 = X_scaled.dot(vectors.T[0])
projected_2 = X_scaled.dot(vectors.T[1])
res = pd.DataFrame(projected_1, columns=['PC1'])
res['PC2'] = projected_2
res['Y'] = y
res.head()
sns.FacetGrid(res, hue="Y", height=6).map(plt.scatter, 'PC1',
'PC2').add_legend()
plt.show()
```