In-Lab

Task 1:

```
# Load libraries
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import
Decision
from sklearn.model_selection import train_test_split # Import
tra
from sklearn import metrics #Import scikit-learn metrics
module f


# Load Dataset
# https://www.kaggle.com/datasets/uciml/pima-indians-
diabetes-database
col_names = ['pregnancies', 'glucose', 'bloodpressure',
'skinthickness', 'insulin', 'bmi', 'pedigree', 'age',
'outcome'] # load dataset
pima = pd.read_csv("diabetes.csv", header=None,
names=col_names)
pima_df= pima.head()
print (pima_df)
```

Output:

```
   pregnancies  glucose  bloodpressure  skinthickness  insulin   bmi  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1
```

	pedigree	age	outcome
0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
4	2.288	33	1

Task 2:

```
#Feature Selection
#
#split dataset in features and target variable
feature_cols = ['pregnancies', 'glucose', 'bloodpressure',
'insulin', 'bmi', 'pedigree','age']
X = pima[feature_cols] # Features
y = pima.outcome # Target variable

#Splitting Data
#Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=1)
#70% training and 30% test



#Create Decision Tree classifer object
clf = DecisionTreeClassifier()

#Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

y_pred = clf.predict(X_test)
```

Task 3:

```
#Model Accuracy, how often is the classifier correct?
print("Accuracy: ", metrics.accuracy_score (y_test,y_pred))
```

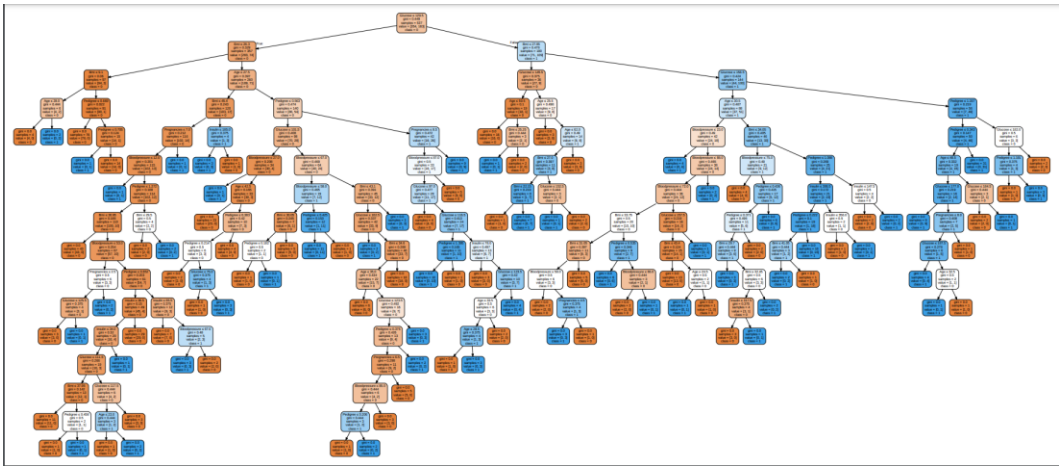
Output:

```
Accuracy:  0.6926406926406926
```

Task 4:

```
#Visualizing Decision Trees
from sklearn.tree import export_graphviz
import graphviz
#Export the decision tree to DOT format
dot_data = export_graphviz(clf, out_file=None,
                           feature_names=X_train.columns, #
Replace with your feature names
                           class_names=[str(x) for x in
clf.classes_], # Convert class names to stri
                           filled=True, rounded=True,
special_characters=True)
#Create and display the graph
graph =graphviz.Source(dot_data)
graph.render("decision_tree") # Saves the visualization as a
file (e.g., "decision_tree.pdf")
graph.view("decision_tree") # Opens the visualization using
the default viewer
```

Output:



Task 5:

```
#Optimizing Decision Tree Performance
# Create Decision Tree classifer object
clf = DecisionTreeClassifier(criterion="entropy",
max_depth=3)
#Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)
#Predict the response for test dataset
y_pred = clf.predict(X_test)
#Model Accuracy, how often is the classifier correct?
print("Accuracy: ",metrics.accuracy_score (y_test,y_pred))
```

Output:

```
Accuracy:  0.7705627705627706
```