```
#Program 1
#Build a Logistic Regression model using all the variables. Use 75% of the data as the training set and fix the random state as 2. The ac

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Read the CSV file into a DataFrame
df = pd.read_csv("People Charm case.csv")

# Define the feature variables (all columns except the target variable)
X = df.drop(columns=['dept','salary'])  # Replace 'target_variable' with the actual target column name

# Define the target variable
y = df['avgMonthlyHours']  # Replace 'target_variable' with the actual target column name

# Split the data into training and testing sets (75% training, 25% testing) with a fixed random state
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=2)

# Initialize the Logistic Regression model
model = LogisticRegression()

# Fit the model on the training data
model.fit(X_train, y_train)

# Predict the target values on the testing data
y_pred = model.predict(X_test)

# Calculate the accuracy score
accuracy = accuracy_score(y_test, y_pred)

print("Accuracy score:", accuracy)
```

```
Accuracy score: 0.017066666666666667
C:\Users\DELL\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (stat
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

```
#Program 2
#Build a Logistic Regression model using all the variables. Use 75% of the data as the training set and fix the random state as 2 and fin

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix

# Read the CSV file into a DataFrame
df = pd.read_csv("People Charm case.csv")

# Define the feature variables (all columns except the target variable)
X = df.drop(columns=['dept','salary'])  # Replace 'target_variable' with the actual target column name

# Define the target variable
y = df['workAccident']  # Replace 'target_variable' with the actual target column name

# Split the data into training and testing sets (75% training, 25% testing) with a fixed random state
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=2)

# Initialize the Logistic Regression model
model = LogisticRegression()

# Fit the model on the training data
model.fit(X_train, y_train)

# Predict the target values on the testing data
y_pred = model.predict(X_test)

# Calculate the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Calculate the number of misclassified samples (sum of off-diagonal elements in the confusion matrix)
misclassified_samples = conf_matrix[0, 1] + conf_matrix[1, 0]
```

```
print("Number of misclassified samples:", misclassified_samples)
```

```
Number of misclassified samples: 0
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

```
#Program 3
# Build a k-Nearest Neighbors model using all the variables. Use 75% of the data as the training set, fix the random state as 0 and the k

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

# Read the CSV file into a DataFrame
df = pd.read_csv("People Charm case.csv")

# Define the feature variables (all columns except the target variable)
X = df.drop(columns=['dept','salary'])  # Replace 'target_variable' with the actual target column name

# Define the target variable
y = df['workAccident']  # Replace 'target_variable' with the actual target column name

# Split the data into training and testing sets (75% training, 25% testing) with a fixed random state
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)

# Initialize the KNN model with k=2
knn_model = KNeighborsClassifier(n_neighbors=2)

# Fit the model on the training data
knn_model.fit(X_train, y_train)

# Predict the target values on the testing data
y_pred = knn_model.predict(X_test)

# Calculate the accuracy score
accuracy = accuracy_score(y_test, y_pred)

print("Accuracy score:", accuracy)
```

```
Accuracy score: 0.9338666666666666
```