Internship project credit card fraud detection

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# 1. Load data
data = pd.read_excel(r'D:\Samsom - All Data\Naresh IT Institute\New folder\default of credit card clients.xls',header=1)

# Drop ID, separate features and target
x = data.drop(['ID', 'default payment next month'], axis=1)
y = data['default payment next month']

# 2. Features and target
x = data.iloc[:, :-1] # Features
y = data.iloc[:, -1] # Target

# 3. Split the data
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_state=42)
```

```
# 4. Standardize
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)

# 5. Models
models = {
    'Logistic Regression': LogisticRegression(max_iter=1000),
    'KNN-Classifier': KNeighborsClassifier(),
    'SVM': SVC()
}

# 6. Train & Evaluate
accuracies = [] # <-- Define the list to store results

for name, model in models.items():
    print(f"Model: {name}")
    model.fit(x_train, y_train)
    y_pred = model.predict(x_test)
    acc = accuracy_score(y_test, y_pred)
    accuracies.append((name, acc)) # <-- Append results here
    print(f"Accuracy: {acc:.4f}")
    print("Classification Report:")
    print(classification_report(y_test, y_pred))
    print("-" * 50)

# 7. Plot Accuracy Comparison
model_names = [i[0] for i in accuracies]
```

```
accuracy_values = [i[1] for i in accuracies]


plt.figure(figsize=(8, 5))
plt.bar(model_names, accuracy_values, color=['skyblue', 'lightgreen', 'salmon'])
plt.title("Model Accuracy Comparison")
plt.xlabel("Model")
plt.ylabel("Accuracy")
plt.ylim(0, 1)
plt.grid(True, linestyle="--", alpha=0.6)
for i, v in enumerate(accuracy_values):
    plt.text(i, v + 0.01, f"{v:.2f}", ha='center', fontweight='bold')
plt.tight_layout()
plt.show()
```