In [34]:
```
import numpy as np
import pandas as pd
from sklearn.neural_network import MLPClassifier
from sklearn.svm import LinearSVC
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
import matplotlib.pyplot as plt
```

In [3]:
```
dataset_train = pd.read_csv('./dataset/mnist/mnist_train.csv')
dataset_test = pd.read_csv('./dataset/mnist/mnist_test.csv')

X_train = dataset_train.drop('label', axis=1)
y_train = dataset_train['label']

X_test = dataset_test.drop('label', axis=1)
y_test = dataset_test['label']
```

In [4]:
```
#MLP
mlp_model = MLPClassifier(hidden_layer_sizes = (30,), activation = 'relu', max_iter = 20
mlp_model.fit(X_train, y_train)
y_pred_mlp = mlp_model.predict(X_test)

acc_mlp = accuracy_score(y_test, y_pred_mlp)
precision_mlp = precision_score(y_test, y_pred_mlp, average='macro')
recall_mlp = recall_score(y_test, y_pred_mlp, average='macro')
f1_mlp = f1_score(y_test, y_pred_mlp, average='macro')
```

```
D:\ProgramData\anaconda3\lib\site-packages\sklearn\neural_network\_multilayer_perceptro
n.py:684: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (20) reached and
the optimization hasn't converged yet.
  warnings.warn(
```

In [19]:
```
#SVM
svm_model = LinearSVC()
svm_model.fit(X_train, y_train)
y_pred_svm = svm_model.predict(X_test)

acc_svm = accuracy_score(y_test, y_pred_svm)
precision_svm = precision_score(y_test, y_pred_svm, average='macro')
recall_svm = recall_score(y_test, y_pred_svm, average='macro')
f1_svm = f1_score(y_test, y_pred_svm, average='macro')
```

```
D:\ProgramData\anaconda3\lib\site-packages\sklearn\svm\_base.py:1244: ConvergenceWarnin
g: Liblinear failed to converge, increase the number of iterations.
  warnings.warn(
```

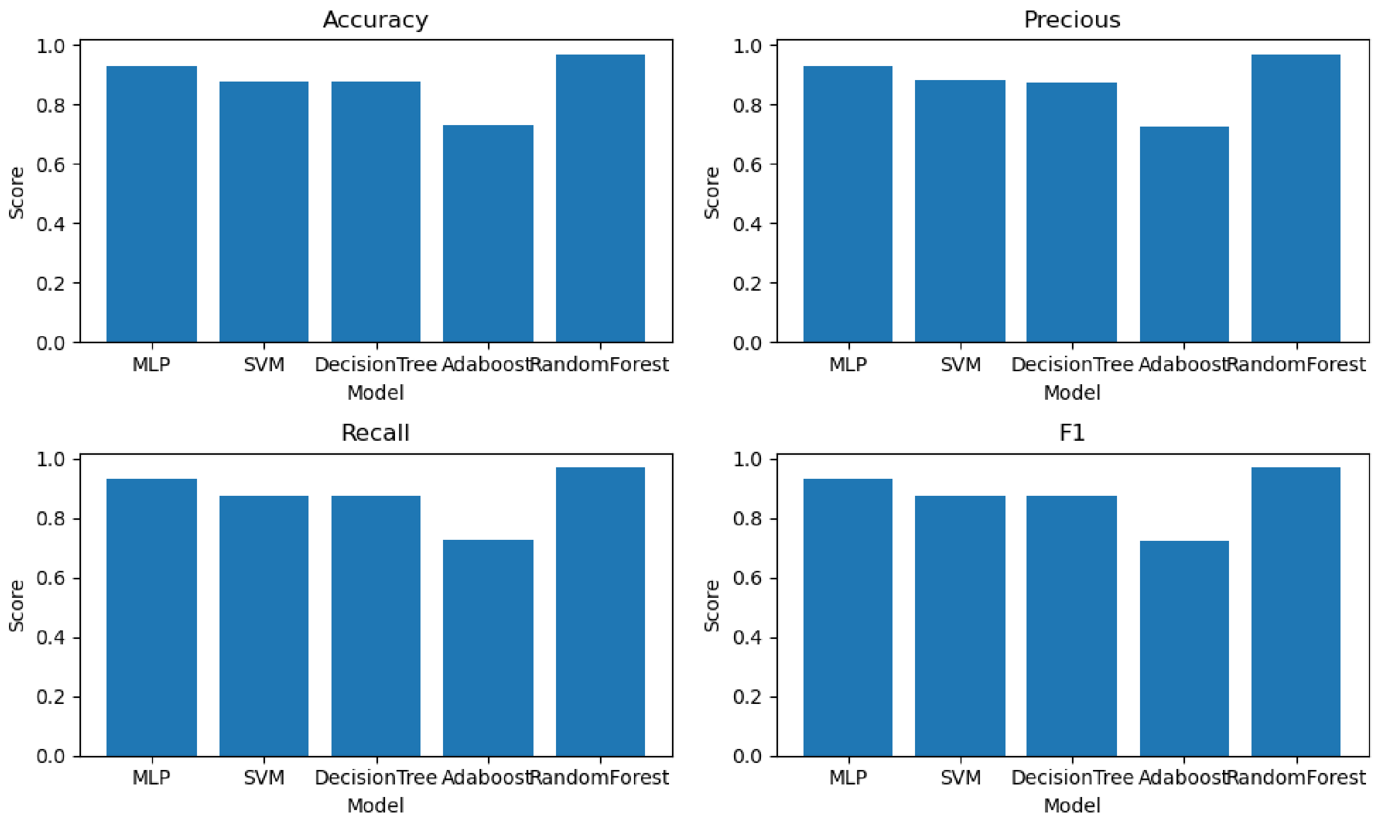
In [25]:
```
#DecisionTree
dt_model = DecisionTreeClassifier(random_state = 42)
dt_model.fit(X_train, y_train)
y_pred_dt = dt_model.predict(X_test)

acc_dt = accuracy_score(y_test, y_pred_dt)
precision_dt = precision_score(y_test, y_pred_dt, average='macro')
recall_dt = recall_score(y_test, y_pred_dt, average='macro')
f1_dt = f1_score(y_test, y_pred_dt, average='macro')
```

In [28]:
```
#AdaBoost
ada_model = AdaBoostClassifier(random_state = 42)
ada_model.fit(X_train, y_train)
y_pred_ada = ada_model.predict(X_test)
```

```python
acc_ada = accuracy_score(y_test, y_pred_ada)
precision_ada = precision_score(y_test, y_pred_ada, average='macro')
recall_ada = recall_score(y_test, y_pred_ada, average='macro')
f1_ada = f1_score(y_test, y_pred_ada, average='macro')
```

In [30]:
```python
#RandomForest
rf_model = RandomForestClassifier(random_state = 42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)

acc_rf = accuracy_score(y_test, y_pred_rf)
precision_rf = precision_score(y_test, y_pred_rf, average='macro')
recall_rf = recall_score(y_test, y_pred_rf, average='macro')
f1_rf = f1_score(y_test, y_pred_rf, average='macro')
```

In [35]:
```python
models = ['MLP', 'SVM', 'DecisionTree', 'Adaboost', 'RandomForest']
accuracy_scores = [acc_mlp, acc_svm, acc_dt, acc_ada, acc_rf]
precision_scores = [precision_mlp, precision_svm, precision_dt, precision_ada, precision
recall_scores = [recall_mlp, recall_svm, recall_dt, recall_ada, recall_rf]
f1_scores = [f1_mlp, f1_svm, f1_dt, f1_ada, f1_rf]

plt.figure(figsize=(10, 6))

plt.subplot(2, 2, 1)
plt.bar(models, accuracy_scores)
plt.title('Accuracy')
plt.xlabel('Model')
plt.ylabel('Score')

plt.subplot(2, 2, 2)
plt.bar(models, precision_scores)
plt.title('Precious')
plt.xlabel('Model')
plt.ylabel('Score')

plt.subplot(2, 2, 3)
plt.bar(models, recall_scores)
plt.title('Recall')
plt.xlabel('Model')
plt.ylabel('Score')

plt.subplot(2, 2, 4)
plt.bar(models, f1_scores)
plt.title('F1')
plt.xlabel('Model')
plt.ylabel('Score')

plt.tight_layout()
plt.show()
```



在默认参数下 多层感知机和随机森林算法表现出惊人的能力