In [1]:
```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
```

In [2]:
```python
# Read in the data from the CSV file
df = pd.read_csv('C:\\Users\\sahil\\OneDrive\\Desktop\\Fall 2020\\Security Analytics\\LR\\datasets\\pay
ment_fraud.csv')
pd.set_option('display.max_columns', None)
```

In [3]:
```python
df.sample (30)
```

Out[3]:

	accountAgeDays	numItems	localTime	paymentMethod	paymentMethodAgeDays	label
28527	167	1	4.461622	paypal	166.626389	0
10827	1284	1	4.876771	creditcard	465.113889	0
22883	34	2	4.921318	creditcard	27.488889	0
9263	103	2	4.921318	creditcard	17.054167	0
8044	1571	1	4.886641	creditcard	9.829861	0
3400	2	1	4.524580	creditcard	1.734722	0
4445	953	1	3.954522	creditcard	71.624306	0
9723	3	1	4.962055	creditcard	0.710417	0
9931	240	1	4.836982	creditcard	0.000000	0
25988	164	1	4.895263	creditcard	0.000694	0
29221	21	1	4.895263	creditcard	0.000694	0
23955	1299	1	4.886641	paypal	88.975000	0
29323	542	1	4.742303	creditcard	474.705556	0
5618	1813	1	4.057414	creditcard	1058.118750	0
4588	2000	1	4.748314	creditcard	533.863889	0
25351	184	1	4.461622	creditcard	43.979861	0
10951	2000	1	4.965339	paypal	0.000000	0
4221	1640	1	4.895263	storecredit	246.623611	0
20676	2000	1	4.921318	creditcard	0.038194	0
4017	5	2	4.895263	creditcard	4.922917	0
12266	1162	1	4.965339	creditcard	85.881250	0
20421	2000	1	4.836982	paypal	0.000694	0
1899	1382	1	5.017904	creditcard	2.979861	0
1750	2000	1	4.748314	creditcard	2.069444	0
38039	803	1	4.745402	creditcard	0.000000	0
35514	2	1	4.057414	creditcard	1.468750	0
14790	2	1	4.876771	paypal	1.783333	0
25148	16	1	4.962055	paypal	15.005556	0
17830	219	1	5.034622	storecredit	0.000694	0
10694	2000	1	5.034622	creditcard	3.093750	0

In [3]:
```python
# Convert categorical feature into dummy variables with one-hot encoding

df = pd.get_dummies(df, columns=['paymentMethod'])
df.sample(3)
```

Out[3]:

	accountAgeDays	numItems	localTime	paymentMethodAgeDays	label	paymentMethod_creditcard	paymentMethod_paypal	paymen
9515	141	1	5.040929	0.000000	0	1	0	
36196	2000	1	4.895263	600.989583	0	0	0	
20937	908	1	5.040929	1.593750	0	1	0	

In [4]:
```python
# Split dataset up into train and test sets,random_state is the seed for random number generator
X_train, X_test, y_train, y_test = train_test_split(
    df.drop('label', axis=1), df['label'],
    test_size=0.33, random_state=133)
```

In [5]:
```python
X_test
```

Out[5]:

	accountAgeDays	numItems	localTime	paymentMethodAgeDays	paymentMethod_creditcard	paymentMethod_paypal	paymentMetho
399	2000	1	5.040929	99.037500	1	0	
4430	90	1	5.034622	0.000000	0	1	
38185	2000	1	4.836982	53.612500	0	0	
30764	481	3	4.921349	33.902778	1	0	
5697	54	1	4.962055	0.000000	1	0	
...	...	...	...	...	...	...	
11076	84	1	4.057414	83.170833	1	0	
36749	24	1	4.524580	0.000694	1	0	
20562	2000	1	4.921318	600.691667	1	0	
30459	2000	1	4.057414	210.464583	1	0	
17173	243	1	4.748314	0.000000	1	0	

12943 rows × 7 columns

In [6]:
```python
# Initialize and train classifier model
clf = LogisticRegression().fit(X_train, y_train)

# Make predictions on test set
y_pred = clf.predict(X_test)
y_score2 = clf.predict_proba(X_test)[:,1]
```

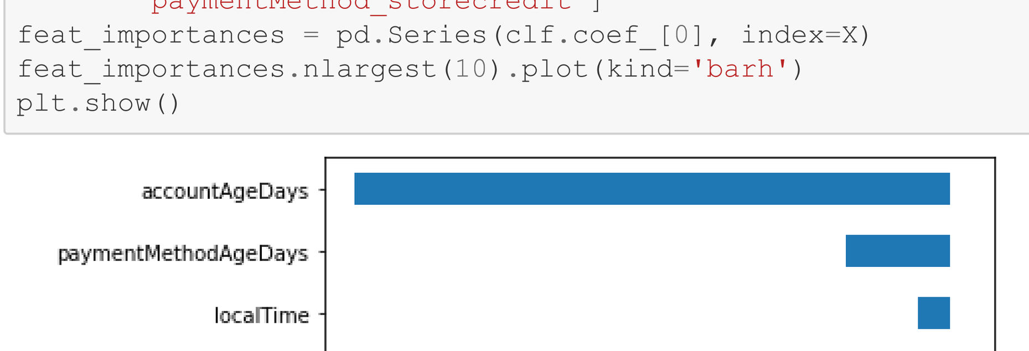
In [7]:
```python
# Compare test set predictions with ground truth labels
print(accuracy_score(y_pred, y_test))
print(confusion_matrix(y_test, y_pred))
```

```
1.0
[[12753     0]
 [    0   190]]
```

In [8]:
```python
df.columns
print(clf.coef_[0])
```

```
[-7.86046486  0.18935419 -0.41173441 -1.38585765  0.02120871  0.02578744
 -0.04703019]
```

In [9]:
```python
X = ['accountAgeDays', 'numItems', 'localTime', 'paymentMethodAgeDays',
     'paymentMethod_creditcard', 'paymentMethod_paypal',
       'paymentMethod_storecredit']
feat_importances = pd.Series(clf.coef_[0], index=X)
feat_importances.nlargest(10).plot(kind='barh')
plt.show()
```



In [15]:
```python
from sklearn.model_selection import KFold

model_accuracy = []
kf = KFold(n_splits=10)
#split train data to train and validation
for train, val in kf.split(X_train, y_train):
    clf = LogisticRegression().fit(X_train.iloc[train], y_train.iloc[train])
    y_pred = clf.predict(X_train.iloc[val])
    model_accuracy.append({'model': clf, 'acc': accuracy_score(y_pred, y_train.iloc[val])})
    print(accuracy_score(y_pred, y_train.iloc[val]))

#choosing the highest accuracy model
clf_best = max(model_accuracy, key = lambda x: x['acc'])['model']

#run it on the test set
y_test_pred = clf_best.predict(X_test)
print('test accuracy:',accuracy_score(y_test, y_test_pred))
```

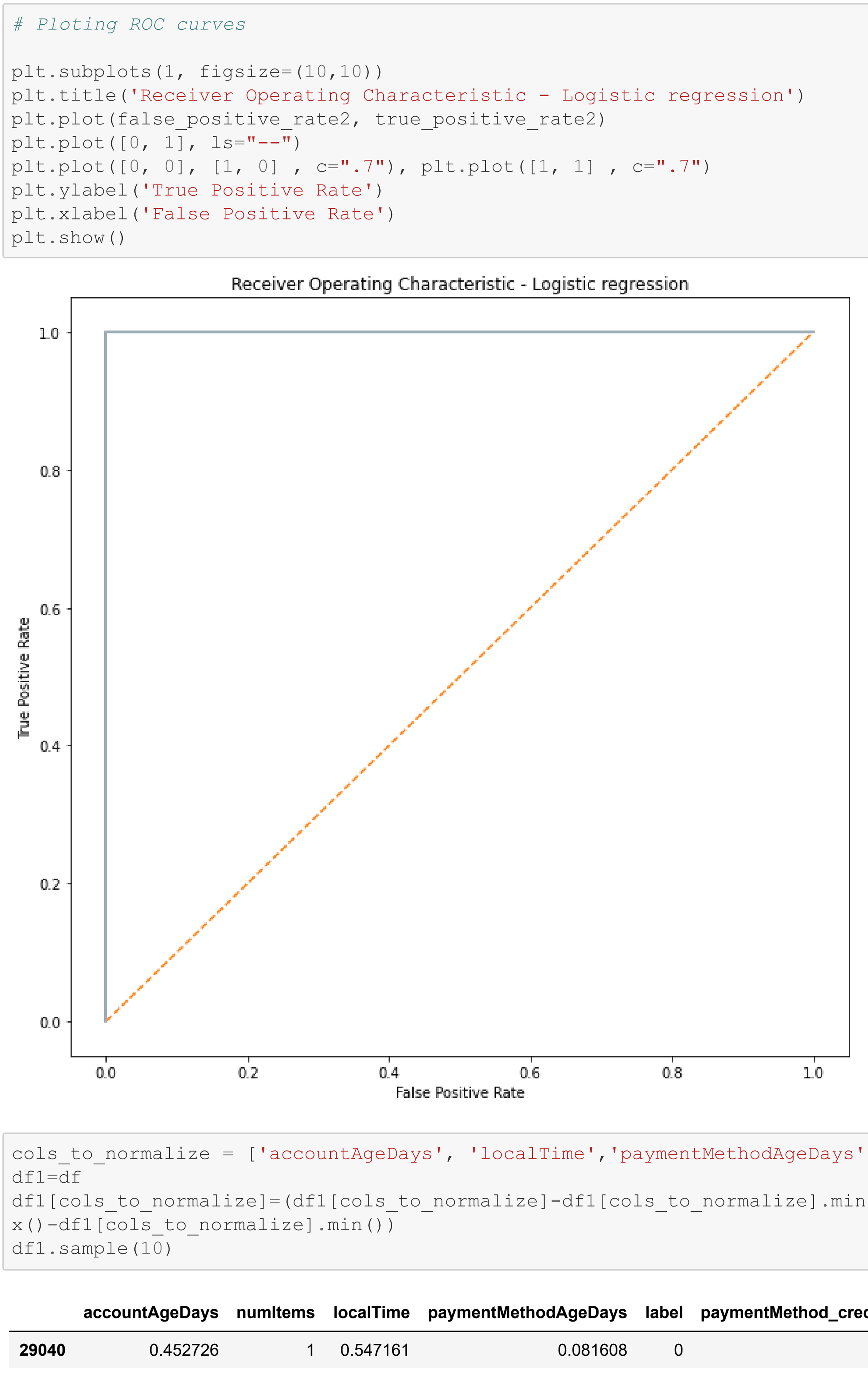
```
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
test accuracy: 1.0
```

In [18]:
```python
#feature selection
```

In [16]:
```python
false_positive_rate2, true_positive_rate2, threshold2 = roc_curve(y_test, y_score2)
print('roc_auc_score for Logistic Regression: ', roc_auc_score(y_test, y_score2))
```

```
roc_auc_score for Logistic Regression:  1.0
```

In [17]:
```python
# Ploting ROC curves

plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic - Logistic regression')
plt.plot(false_positive_rate2, true_positive_rate2)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



In [6]:
```python
cols_to_normalize = ['accountAgeDays', 'localTime','paymentMethodAgeDays']
df1=df
df1[cols_to_normalize]=(df1[cols_to_normalize]-df1[cols_to_normalize].min())/(df1[cols_to_normalize].ma
x()-df1[cols_to_normalize].min())
df1.sample(10)
```

Out[6]:

	accountAgeDays	numItems	localTime	paymentMethodAgeDays	label	paymentMethod_creditcard	paymentMethod_paypal	paymen
29040	0.452726	1	0.547161	0.081608	0	1	0	
7721	0.001001	1	0.888229	0.000835	0	0	1	
35166	0.912956	1	0.974109	0.000000	0	1	0	
13093	0.002001	1	0.982927	0.000000	0	0	1	
23517	0.478739	1	0.974109	0.000689	0	1	0	
28801	1.000000	1	0.995016	0.432763	0	1	0	
25348	1.000000	1	0.974115	0.127477	0	1	0	
22039	0.585793	1	0.936660	0.000000	0	1	0	
32876	0.317159	1	0.964466	0.317429	0	1	0	
18833	0.007504	1	0.968469	0.000000	0	1	0	

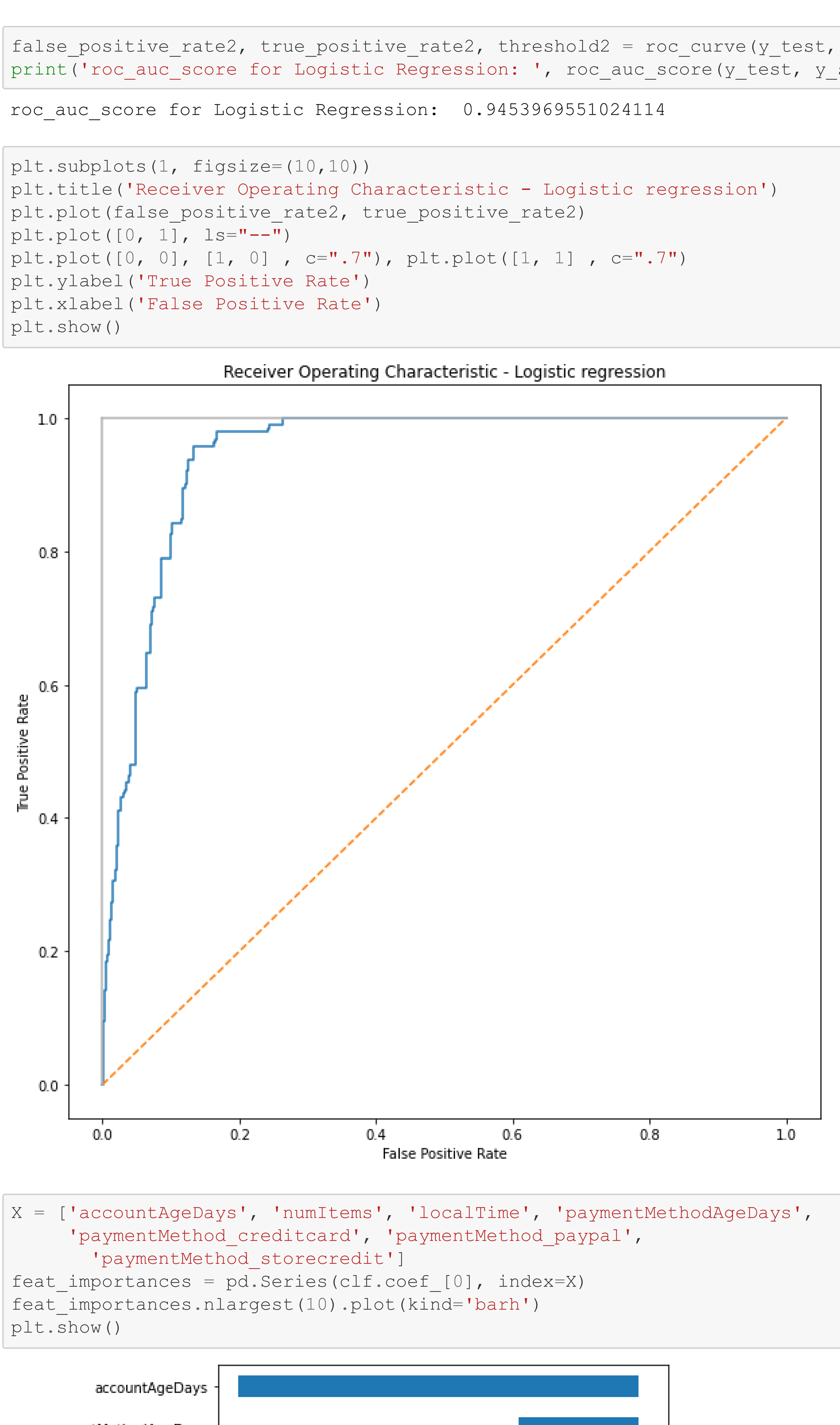
In [7]:
```python
# Split dataset up into train and test sets
X_train, X_test, y_train, y_test = train_test_split(
    df.drop('label', axis=1), df1['label'],
    test_size=0.33, random_state=17)

# Initialize and train classifier model
clf = LogisticRegression().fit(X_train, y_train)

# Make predictions on test set
y_pred = clf.predict(X_test)
y_score2 = clf.predict_proba(X_test)[:,1]

# Compare test set predictions with ground truth labels
print(accuracy_score(y_pred, y_test))
print(confusion_matrix(y_test, y_pred))
```

```
0.9852429884879857
[[12752     1]
 [  190     0]]
```

In [9]:
```python
false_positive_rate2, true_positive_rate2, threshold2 = roc_curve(y_test, y_score2)
print('roc_auc_score for Logistic Regression: ', roc_auc_score(y_test, y_score2))
```

```
roc_auc_score for Logistic Regression:  0.9453969551024114
```

In [10]:
```python
plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic - Logistic regression')
plt.plot(false_positive_rate2, true_positive_rate2)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



In [11]:
```python
X = ['accountAgeDays', 'numItems', 'localTime', 'paymentMethodAgeDays',
     'paymentMethod_creditcard', 'paymentMethod_paypal',
       'paymentMethod_storecredit']
feat_importances = pd.Series(clf.coef_[0], index=X)
feat_importances.nlargest(10).plot(kind='barh')
plt.show()
```