```
from google.colab import files
uploaded = files.upload()
```


Choose Files amex.csv

- **amex.csv**(application/vnd.ms-excel) - 20724257 bytes, last modified: 10/4/2019 - 100% done

```
Saving amex.csv to amex.csv
```

```
import io
import pandas as pd
df2 = pd.read_csv(io.BytesIO(uploaded['amex.csv']))
```

```
df2.head()
```


	id	campaign_id	coupon_id	customer_id	redemption_status	type	campaign_type	s
0	1	13	27	1053	0.0	train	0	2
1	2	13	116	48	0.0	train	0	2
2	7	13	644	1050	0.0	train	0	2
3	21	13	1028	89	0.0	train	0	2
4	23	13	517	1067	0.0	train	0	2

```
train = df2.loc[df2['type']=='train']
test = df2.loc[df2['type']=='test']
```

```
X_train=train.drop(['type','start_date','end_date','redemption_status'],axis=1)
Y_train=train['redemption_status']
X_test=test.drop(['type','start_date','end_date','redemption_status'],axis=1)
```

```
X=X_train
Y=Y_train
```

```
from sklearn.model_selection import StratifiedKFold, cross_val_score,GridSearchCV
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier, BaggingClassifier, AdaBoostClassifier, Gradi
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier

cv=StratifiedKFold(n_splits=5,random_state=5)

models = []
models.append(('CART', DecisionTreeClassifier()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('Naive Bayes', GaussianNB()))
models.append(('SVM', SVC()))
```

```
models.append(('Random Forest', RandomForestClassifier()))
models.append(('Bagging', BaggingClassifier()))
models.append(('AdaBoost', AdaBoostClassifier()))
models.append(('Gradient Boosting', GradientBoostingClassifier()))
models.append(('Logistic Regression', LogisticRegression()))
models.append(('MLP', MLPClassifier ( max_iter=1000)))

results = []
names = []
final_scores=[]
for name, model in models:
    accuracy=cross_val_score(model, X, Y,cv=cv)
    results.append(accuracy)
    names.append(name)
    score_mean = "%s: %f" % (name, accuracy.mean())
    final_scores.append(score_mean)
final_scores
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/svm/base.py:193: FutureWarning: The de
  "avoid this warning.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/svm/base.py:193: FutureWarning: The de
  "avoid this warning.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/svm/base.py:193: FutureWarning: The de
  "avoid this warning.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/svm/base.py:193: FutureWarning: The de
  "avoid this warning.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/svm/base.py:193: FutureWarning: The de
  "avoid this warning.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/ensemble/forest.py:245: FutureWarning:
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/ensemble/forest.py:245: FutureWarning:
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/ensemble/forest.py:245: FutureWarning:
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/ensemble/forest.py:245: FutureWarning:
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/ensemble/forest.py:245: FutureWarning:
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: FutureWa
  FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: FutureWa
  FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: FutureWa
  FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: FutureWa
  FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: FutureWa
  FutureWarning)
['CART: 0.788452',
 'KNN: 0.990660',
 'Naive Bayes: 0.981727',
 'SVM: 0.990698',
 'Random Forest: 0.883310',
 'Bagging: 0.834643',
 'AdaBoost: 0.900140',
 'Gradient Boosting: 0.822713',
 'Logistic Regression: 0.990698',
 'MLP: 0.986806']
```