In [70]:
```
#code along with confusion matrix

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix

from sklearn.neighbors import KNeighborsClassifier

from sklearn.svm import SVC


from sklearn import metrics

from mlxtend.plotting import plot_confusion_matrix
```

In [71]:
```
# Step 2: Import the dataset provided
loan_df = pd.read_csv("loan-predictionUC.csv")
loan_df.drop(['Loan_ID'],axis=1,inplace=True)
loan_df
```

Out[71]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
0	Male	No	0.0	Graduate	No	5849	
1	Male	Yes	1.0	Graduate	No	4583	
2	Male	Yes	0.0	Graduate	Yes	3000	
3	Male	Yes	0.0	Not Graduate	No	2583	
4	Male	No	0.0	Graduate	No	6000	
...	...	...	...	...	...	...	
593	Female	No	0.0	Graduate	No	2900	
594	Male	Yes	3.0	Graduate	No	4106	
595	Male	Yes	1.0	Graduate	No	8072	
596	Male	Yes	2.0	Graduate	No	7583	
597	Female	No	0.0	Graduate	Yes	4583	

598 rows × 12 columns

In [72]:
```
#Transforming values in digits
label_encoder = LabelEncoder()
obj=(loan_df.dtypes=='object')
for col in list(obj[obj].index):
```

```
    loan_df[col]=label_encoder.fit_transform(loan_df[col])
loan_df
```

Out[72]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
0	1	0	0.0	0	0	5849	
1	1	1	1.0	0	0	4583	
2	1	1	0.0	0	1	3000	
3	1	1	0.0	1	0	2583	
4	1	0	0.0	0	0	6000	
...	...	...	...	...	...	...	
593	0	0	0.0	0	0	2900	
594	1	1	3.0	0	0	4106	
595	1	1	1.0	0	0	8072	
596	1	1	2.0	0	0	7583	
597	0	0	0.0	0	1	4583	

598 rows × 12 columns

In [73]:
```
#Null values
loan_df.isna().sum()
```

Out[73]:
```
Gender               0
Married              0
Dependents          12
Education            0
Self_Employed        0
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount          21
Loan_Amount_Term    14
Credit_History      49
Property_Area        0
Loan_Status          0
dtype: int64
```

In [74]:
```
#After filling Null
for col in loan_df.columns:
    loan_df[col]=loan_df[col].fillna(loan_df[col].mean())
loan_df
```

Out[74]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
0	1	0	0.0	0	0	5849	
1	1	1	1.0	0	0	4583	
2	1	1	0.0	0	1	3000	
3	1	1	0.0	1	0	2583	
4	1	0	0.0	0	0	6000	
...	...	...	...	...	...	...	
593	0	0	0.0	0	0	2900	
594	1	1	3.0	0	0	4106	
595	1	1	1.0	0	0	8072	
596	1	1	2.0	0	0	7583	
597	0	0	0.0	0	1	4583	

598 rows × 12 columns

In [75]:
```
#Splitting Dataset
X =loan_df.drop(['Loan_Status'],axis=1)
Y = loan_df['Loan_Status']

X.shape,Y.shape

X_train, X_test, Y_train, Y_test = train_test_split(X, Y,test_size=0.4,random_state
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

Out[75]:
```
((358, 11), (240, 11), (358,), (240,))
```

In [76]:
```
#Model Training and Evaluation
knn = KNeighborsClassifier(n_neighbors=3)
rfc = RandomForestClassifier(n_estimators = 7,criterion = 'entropy',random_state =7
svc = SVC()
#lc = LogisticRegression()

# making predictions on the training set
for clf in (rfc, knn, svc):
        clf.fit(X_train, Y_train)
        Y_pred = clf.predict(X_train)
        print("Evaluation score of ",clf.__class__.__name__,"=",100*metrics.accurac
```

```
Evaluation score of  RandomForestClassifier = 98.04469273743017
Evaluation score of  KNeighborsClassifier = 78.49162011173185
Evaluation score of  SVC = 68.71508379888269
```

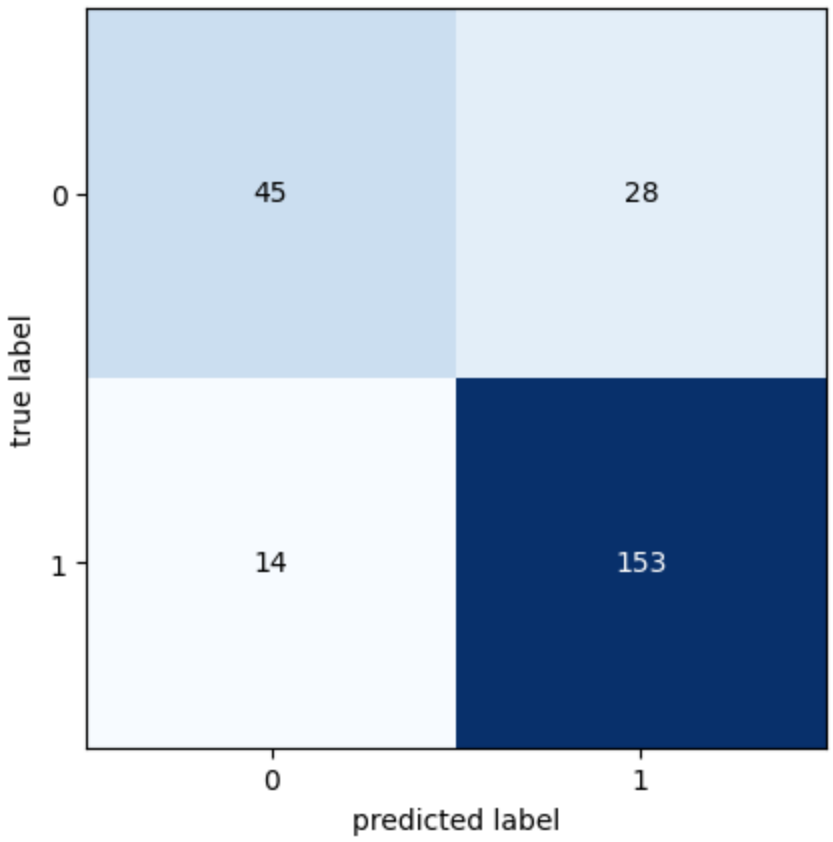
In [77]:
```
# making predictions on the testing set
for clf in (rfc, knn, svc):
        clf.fit(X_train, Y_train)
```

```
        Y_pred = clf.predict(X_test)
        print("Accuracy score of ",clf.__class__.__name__,"=",100*metrics.accuracy_
```

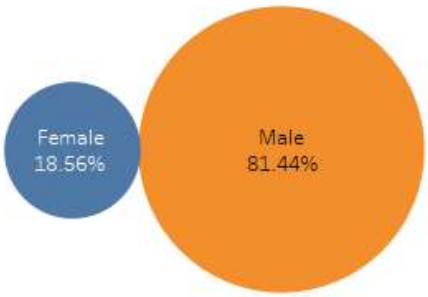
```
Accuracy score of  RandomForestClassifier = 82.5
Accuracy score of  KNeighborsClassifier = 63.74999999999999
Accuracy score of  SVC = 69.16666666666667
```

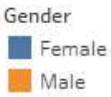
In [78]:
```
# we will perform testing of the models using the test set of data
print("Highest accuracy: ")
rfc.fit(X_train, Y_train)
Y_pred = rfc.predict(X_test)
cm = confusion_matrix(Y_test,Y_pred)
plot_confusion_matrix(cm)
print("Accuracy score of ",rfc.__class__.__name__,"=",100*metrics.accuracy_score(Y_
```

```
Highest accuracy:
Accuracy score of  RandomForestClassifier = 82.5
```

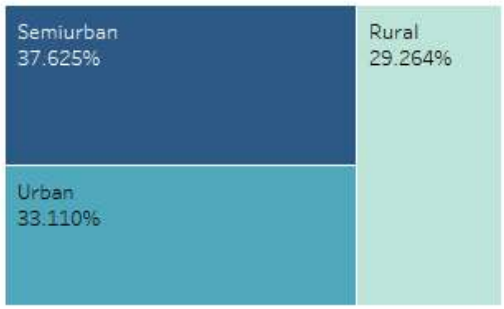


gender vs Loan Approval





Area vs loan status



Loan status,Property area vs Loan approval