Import Library

In [40]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.compose import make_column_transformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import MinMaxScaler
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import GridSearchCV
```

Data Preprocessing

In [2]:
```
bank=pd.read_csv('bank-full.csv',sep=';')
bank
```

Out[2]:

	age	job	marital	education	default	balance	housing	loan	contact	day	n
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	
1	44	technician	single	secondary	no	29	yes	no	unknown	5	
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	
4	33	unknown	single	unknown	no	1	no	no	unknown	5	
...	...	...	...	...	...	...	...	...	...	...	
45206	51	technician	married	tertiary	no	825	no	no	cellular	17	
45207	71	retired	divorced	primary	no	1729	no	no	cellular	17	
45208	72	retired	married	secondary	no	5715	no	no	cellular	17	
45209	57	blue-collar	married	secondary	no	668	no	no	telephone	17	
45210	37	entrepreneur	married	secondary	no	2971	no	no	cellular	17	

45211 rows × 17 columns

In [3]:
```
bank.info()
print("\n\n Data pada kolom y= ",bank['y'].unique())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        45211 non-null  int64 
 1   job        45211 non-null  object
 2   marital    45211 non-null  object
 3   education  45211 non-null  object
 4   default    45211 non-null  object
 5   balance    45211 non-null  int64 
 6   housing    45211 non-null  object
 7   loan       45211 non-null  object
 8   contact    45211 non-null  object
 9   day        45211 non-null  int64 
 10  month      45211 non-null  object
 11  duration   45211 non-null  int64 
 12  campaign   45211 non-null  int64 
 13  pdays      45211 non-null  int64 
 14  previous   45211 non-null  int64 
 15  poutcome   45211 non-null  object
 16  y          45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB


 Data pada kolom y=  ['no' 'yes']
```

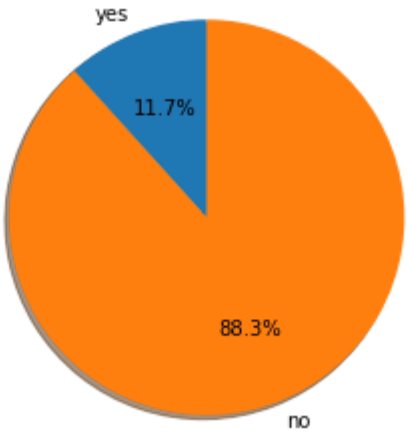
dapat disimpulkan bahwa tidak terdapat missing value,
tipe data untuk numerical feature adalah int64 dan categorical feature adalah object
Klasifikasi yang dilakukan yaitu binary classification karena hanya terdapat 2 nilai pada kolom y yaitu 0 dan 1

In [5]:
```
# imbalance dataset

bank['y'].value_counts()
yesCustomer=len(bank[bank['y']=='yes'])
noCustomer=len(bank[bank['y']=='no'])
totalCustomer=yesCustomer+noCustomer
labels='yes','no'
sizes=[(yesCustomer*100/totalCustomer),(noCustomer*100/totalCustomer)]

fig1, ax1 = plt.subplots()
ax1.pie(sizes, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```



Dari pie chart di atas terlihat bahwa dataset merupakan imbalance dataset.

In [6]:
```
# ubah data kolom y menjadi angka
bank['y']=bank['y'].replace({'yes':1,'no':0})
```

In [7]:
```
# pisah kolom y dan kolom2 fitur(X)
y=bank['y']
X_lama=bank.drop('y',axis='columns')
```

In [8]:
```
print("Dimensi sebelum preprocessing")
print("X= ",X_lama.shape)
print("y= ",y.shape )
```

```
Dimensi sebelum preprocessing
X=  (45211, 16)
y=  (45211,)
```

In [9]:
```
# memisahkan kolom2 yang kategorical dan numerical
numerical_ix = X_lama.select_dtypes(include=['int64']).columns
categorical_ix = X_lama.select_dtypes(include=['object']).columns
```

Data kategorikal akan di-preprocess dengan OneHotEncoder
Data numerikal akan di-preprocess dengan MinMaxScaler

In [13]:
```
t = [('cat', OneHotEncoder(), categorical_ix), ('num', MinMaxScaler(), numeric
al_ix)]
col_transform = ColumnTransformer(transformers=t)
transformed_col=col_transform.fit_transform(X_lama)
X=transformed_col
```

In [14]:
```
print("Dimensi setelah preprocessing")
print("X= ",X.shape)
print("y= ",y.shape)
```

```
Dimensi setelah preprocessing
X=  (45211, 51)
y=  (45211,)
```

In [22]:
```
# cek kesesuaian jumlah kolom fitur hasil preprocessing ->
print("cek kesesuaian jumlah kolom fitur hasil preprocessing \n")
sum=0
for i in categorical_ix:
    print(i,"=",len(X_lama[i].unique())," kategori")
    sum=sum+len(X_lama[i].unique())
print("\nTOTAL KATEGORI= ",sum)
print("TOTAL KOLOM NUMERICAL= ",len(numerical_ix))
print("TOTAL = ",sum+len(numerical_ix))
```

```
cek kesesuaian jumlah kolom fitur hasil preprocessing

job = 12  kategori
marital = 3  kategori
education = 4  kategori
default = 2  kategori
housing = 2  kategori
loan = 2  kategori
contact = 3  kategori
month = 12  kategori
poutcome = 4  kategori

TOTAL KATEGORI=  44
TOTAL KOLOM NUMERICAL=  7
TOTAL =  51
```

Kesimpulan: Jumlah kolom hasil preprocessing sama dengan perhitungan manual yaitu jumlah katogeori + jumlah kolom numerikal

XGBoost

In [24]:
```
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
import time
```

XGBoost default hyperparameter

In [37]:
```
model=XGBClassifier()
```

In [38]:
```
start_time = time.time()
kfold=KFold(n_splits=5)
results=cross_val_score(model,X,y,cv=kfold,scoring="roc_auc")
print(f"AUC: {round(results.mean(),4)}, std: {round(results.std(),4)}")
print(f"Waktu: {round((time.time() - start_time),4)} seconds" )
```

```
AUC: 0.8562, std: 0.0558
Waktu: 12.7012 seconds
```

XGBoost + GridSearch

In [39]:
```
params={
    'eta': [0.001,0.01,0.1], # learning rate
    'subsample': [0.1,0.4,0.8],
    'max_depth': [10,20,30],
    'gamma':[0.1,0.4,0.8],
    'min_child_weight':[2,5,11]
}
```

In [41]:
```
grid_search_clf = GridSearchCV(
    estimator=model,
    param_grid=params,
    scoring = 'roc_auc',
    n_jobs = 10,
    cv = 5,
    verbose=10
)
```

In [42]:
```
grid_search_clf.fit(X,y)
```

```
Fitting 5 folds for each of 243 candidates, totalling 1215 fits

[Parallel(n_jobs=10)]: Using backend LokyBackend with 10 concurrent workers.
[Parallel(n_jobs=10)]: Done   5 tasks      | elapsed:   26.2s
[Parallel(n_jobs=10)]: Done  12 tasks      | elapsed:   46.7s
[Parallel(n_jobs=10)]: Done  21 tasks      | elapsed:  1.2min
[Parallel(n_jobs=10)]: Done  30 tasks      | elapsed:  1.5min
[Parallel(n_jobs=10)]: Done  41 tasks      | elapsed:  2.1min
[Parallel(n_jobs=10)]: Done  52 tasks      | elapsed:  2.9min
[Parallel(n_jobs=10)]: Done  65 tasks      | elapsed:  3.5min
[Parallel(n_jobs=10)]: Done  78 tasks      | elapsed:  4.4min
[Parallel(n_jobs=10)]: Done  93 tasks      | elapsed:  5.4min
[Parallel(n_jobs=10)]: Done 108 tasks      | elapsed:  6.5min
[Parallel(n_jobs=10)]: Done 125 tasks      | elapsed:  7.9min
[Parallel(n_jobs=10)]: Done 142 tasks      | elapsed:  8.7min
[Parallel(n_jobs=10)]: Done 161 tasks      | elapsed:  9.7min
[Parallel(n_jobs=10)]: Done 180 tasks      | elapsed: 10.6min
[Parallel(n_jobs=10)]: Done 201 tasks      | elapsed: 12.3min
[Parallel(n_jobs=10)]: Done 222 tasks      | elapsed: 13.9min
[Parallel(n_jobs=10)]: Done 245 tasks      | elapsed: 15.8min
[Parallel(n_jobs=10)]: Done 268 tasks      | elapsed: 17.9min
[Parallel(n_jobs=10)]: Done 293 tasks      | elapsed: 19.2min
[Parallel(n_jobs=10)]: Done 318 tasks      | elapsed: 20.3min
[Parallel(n_jobs=10)]: Done 345 tasks      | elapsed: 22.5min
[Parallel(n_jobs=10)]: Done 372 tasks      | elapsed: 24.9min
[Parallel(n_jobs=10)]: Done 401 tasks      | elapsed: 27.2min
[Parallel(n_jobs=10)]: Done 430 tasks      | elapsed: 28.7min
[Parallel(n_jobs=10)]: Done 461 tasks      | elapsed: 30.6min
[Parallel(n_jobs=10)]: Done 492 tasks      | elapsed: 32.7min
[Parallel(n_jobs=10)]: Done 525 tasks      | elapsed: 35.1min
[Parallel(n_jobs=10)]: Done 558 tasks      | elapsed: 36.8min
[Parallel(n_jobs=10)]: Done 593 tasks      | elapsed: 38.8min
[Parallel(n_jobs=10)]: Done 628 tasks      | elapsed: 41.2min
[Parallel(n_jobs=10)]: Done 665 tasks      | elapsed: 44.0min
[Parallel(n_jobs=10)]: Done 702 tasks      | elapsed: 45.9min
[Parallel(n_jobs=10)]: Done 741 tasks      | elapsed: 48.4min
[Parallel(n_jobs=10)]: Done 780 tasks      | elapsed: 51.2min
[Parallel(n_jobs=10)]: Done 821 tasks      | elapsed: 53.8min
[Parallel(n_jobs=10)]: Done 862 tasks      | elapsed: 56.1min
[Parallel(n_jobs=10)]: Done 905 tasks      | elapsed: 58.7min
[Parallel(n_jobs=10)]: Done 948 tasks      | elapsed: 61.6min
[Parallel(n_jobs=10)]: Done 993 tasks      | elapsed: 63.8min
[Parallel(n_jobs=10)]: Done 1038 tasks      | elapsed: 67.0min
[Parallel(n_jobs=10)]: Done 1085 tasks      | elapsed: 70.2min
[Parallel(n_jobs=10)]: Done 1132 tasks      | elapsed: 72.8min
[Parallel(n_jobs=10)]: Done 1181 tasks      | elapsed: 76.4min
[Parallel(n_jobs=10)]: Done 1215 out of 1215 | elapsed: 78.6min finished
```

Out[42]:
```
GridSearchCV(cv=5, error_score=nan,
             estimator=XGBClassifier(base_score=None, booster=None,
                                     colsample_bylevel=None,
                                     colsample_bynode=None,
                                     colsample_bytree=None, gamma=None,
                                     gpu_id=None, importance_type='gain',
                                     interaction_constraints=None,
                                     learning_rate=None, max_delta_step=None,
                                     max_depth=None, min_child_weight=None,
                                     missing=nan, monotone_constraints=None,
                                     n_estim...
                                     reg_lambda=None, scale_pos_weight=None,
                                     subsample=None, tree_method=None,
                                     validate_parameters=None, verbosity=Non
e),
             iid='deprecated', n_jobs=10,
             param_grid={'gamma': [0.1, 0.4, 0.8],
                         'learning_rate': [0.001, 0.01, 0.1],
                         'max_depth': [10, 20, 30],
                         'min_child_weight': [2, 5, 11],
                         'subsample': [0.1, 0.4, 0.8]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring='roc_auc', verbose=10)
```

In [46]:
```
grid_search_clf.best_params_
```

Out[46]:
```
{'gamma': 0.4,
 'learning_rate': 0.01,
 'max_depth': 10,
 'min_child_weight': 11,
 'subsample': 0.1}
```

In [48]:
```
grid_search_clf.best_score_
```

Out[48]:
```
0.7382674790807361
```

Nilai AUC pada model yang menggunakan default hyperparameter lebih besar dibandingkan yang menggunakan hyperparamete hasil dari gridsearch. Hal ini sangat mungkin terjadi karena hyperparameter search space pada gridsearch terbatas.

In [ ]:
```

```