Hasil dan Analisa :

- Iterasi 100 dan 50 pada RS menghasilkan nilai AUC yang sama yaitu 0.695. Hanya berbeda sedikit dibandingkan menggunakan GS yaitu 0.738. Waktu komputasi berbeda cukup jauh, untuk GS diperlukan 82 menit dengan nilai AUC 0.738 sedangkan dengan RS diperlukan 22 menit dengan nilai AUC 0.695
- Nilai RS tidak mencapai nilai GS karena nilai hyperparameter dengan AUC tertinggi pada GS tidak masuk ke dalam hyperparameter search space RS

Code:

In [1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import cross_val_score
from xgboost import XGBClassifier
```

In [2]:
```
X_bank=pd.read_csv('dataset/X_bank_preprocessed.csv').to_numpy()
y_bank=pd.read_csv('dataset/y_bank_preprocessed.csv').to_numpy().ravel()
```

Model dan Hyperparameter Space

In [3]:
```
model=XGBClassifier()
```

hyper_space_1

In [38]:
```
hyper_space_1={
    'eta': list(np.linspace(0.001,1,10)), # learning rate
    'subsample': list(np.linspace(0,1,10)),
    'max_depth': list(range(5,50+1,5)),
    'gamma': list(np.linspace(0,1,10)),
    'min_child_weight': list(range(0,15+1))
}
```

In [12]:
```
print('eta : ',params['eta'],'\n') # learning rate
print('subsample : ',params['subsample'],'\n')
print('max_depth : ',params['max_depth'],'\n')
print('gamma : ',params['gamma'],'\n')
print('min_child_weight : ', params['min_child_weight'],'\n')
```

```
eta :  [0.001, 0.112, 0.223, 0.334, 0.445, 0.556, 0.667, 0.778, 0.889, 1.0]

subsample :  [0.0, 0.1111111111111111, 0.2222222222222222, 0.333333333333333
3, 0.4444444444444444, 0.5555555555555556, 0.6666666666666666, 0.777777777777
7777, 0.8888888888888888, 1.0]

max_depth :  [5, 10, 15, 20, 25, 30, 35, 40, 45, 50]

gamma :  [0.0, 0.1111111111111111, 0.2222222222222222, 0.3333333333333333, 0.
4444444444444444, 0.5555555555555556, 0.6666666666666666, 0.7777777777777777,
0.8888888888888888, 1.0]

min_child_weight :  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15]
```

hyper_space_2

In [4]:
```
params2={
    'eta': list(np.linspace(0.001,1,100)), # learning rate
    'subsample': list(np.linspace(0,1,20)),
    'max_depth': list(range(5,50+1,5)),
    'gamma': list(np.linspace(0,1,50)),
    'min_child_weight': list(range(0,15+1))
}
```

In [5]:
```
print('eta : ',params2['eta'],'\n') # learning rate
print('subsample : ',params2['subsample'],'\n')
print('max_depth : ',params2['max_depth'],'\n')
print('gamma : ',params2['gamma'],'\n')
print('min_child_weight : ', params2['min_child_weight'],'\n')
```

```
eta :  [0.001, 0.011090909090909092, 0.021181818181818184, 0.0312727272727272
7, 0.041363636363636366, 0.05145454545454546, 0.06154545454545455, 0.07163636
363636364, 0.08172727272727273, 0.09181818181818183, 0.10190909090909092, 0.1
12, 0.1220909090909091, 0.13218181818181818, 0.14227272727272727, 0.152363636
36363637, 0.16245454545454546, 0.17254545454545456, 0.18263636363636365, 0.19
272727272727275, 0.20281818181818184, 0.2129090909090909, 0.223, 0.2330909090
909091, 0.2431818181818182, 0.25327272727272726, 0.26336363636363636, 0.27345
454545454545, 0.28354545454545454, 0.29363636363636364, 0.30372727272727273,
0.31381818181818183, 0.3239090909090909, 0.334, 0.3440909090909091, 0.3541818
181818182, 0.3642727272727273, 0.3743636363636364, 0.3844545454545455, 0.3945
454545454546, 0.4046363636363637, 0.4147272727272727, 0.4248181818181818, 0.4
349090909090909, 0.445, 0.4550909090909091, 0.4651818181818182, 0.47527272727
27273, 0.4853636363636364, 0.4954545454545455, 0.5055454545454545, 0.51563636
36363637, 0.5257272727272727, 0.5358181818181819, 0.5459090909090909, 0.556,
0.5660909090909091, 0.5761818181818182, 0.5862727272727273, 0.596363636363636
4, 0.6064545454545455, 0.6165454545454546, 0.6266363636363637, 0.636727272727
2728, 0.6468181818181818, 0.6569090909090909, 0.667, 0.6770909090909091, 0.68
71818181818182, 0.6972727272727273, 0.7073636363636364, 0.7174545454545455,
0.7275454545454546, 0.7376363636363636, 0.7477272727272728, 0.757818181818181
8, 0.767909090909091, 0.778, 0.7880909090909092, 0.7981818181818182, 0.808272
7272727274, 0.8183636363636364, 0.8284545454545454, 0.8385454545454546, 0.848
6363636363636, 0.8587272727272728, 0.8688181818181818, 0.878909090909091, 0.8
89, 0.8990909090909092, 0.9091818181818182, 0.9192727272727274, 0.92936363636
36364, 0.9394545454545455, 0.9495454545454546, 0.9596363636363637, 0.96972727
27272728, 0.9798181818181818, 0.989909090909091, 1.0]

subsample :  [0.0, 0.05263157894736842, 0.10526315789473684, 0.15789473684210
525, 0.21052631578947367, 0.2631578947368421, 0.3157894736842105, 0.368421052
6315789, 0.42105263157894735, 0.47368421052631576, 0.5263157894736842, 0.5789
473684210527, 0.631578947368421, 0.6842105263157894, 0.7368421052631579, 0.78
94736842105263, 0.8421052631578947, 0.894736842105263, 0.9473684210526315, 1.
0]

max_depth :  [5, 10, 15, 20, 25, 30, 35, 40, 45, 50]

gamma :  [0.0, 0.02040816326530612, 0.04081632653061224, 0.06122448979591836
6, 0.08163265306122448, 0.1020408163265306, 0.12244897959183673, 0.1428571428
5714285, 0.16326530612244897, 0.18367346938775508, 0.2040816326530612, 0.2244
8979591836732, 0.24489795918367346, 0.26530612244897955, 0.2857142857142857,
0.3061224489795918, 0.32653061224489793, 0.3469387755102041, 0.36734693877551
017, 0.3877551020408163, 0.4081632653061224, 0.42857142857142855, 0.448979591
83673464, 0.4693877551020408, 0.4897959183673469, 0.5102040816326531, 0.53061
22448979591, 0.5510204081632653, 0.5714285714285714, 0.5918367346938775, 0.61
22448979591836, 0.6326530612244897, 0.6530612244897959, 0.673469387755102, 0.
6938775510204082, 0.7142857142857142, 0.7346938775510203, 0.7551020408163265,
0.7755102040816326, 0.7959183673469387, 0.8163265306122448, 0.83673469387755
1, 0.8571428571428571, 0.8775510204081632, 0.8979591836734693, 0.918367346938
7754, 0.9387755102040816, 0.9591836734693877, 0.9795918367346939, 1.0]

min_child_weight :  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
```

fungsi : randomsearch

In [ ]:
```
def randomsearch(X,y,model,parameter,iterasi,evalscore,seed):
    auc=[]
    std=[]

    random_search=RandomizedSearchCV(
                  model,
                  parameter,
                  n_iter=iterasi,
                  scoring=evalscore,
                  n_jobs=-1, cv=5,
                  random_state=seed)
    random_search.fit(X,y)

    best_index=random_search.best_index_
    auc.append(random_search.cv_results_['mean_test_score'])
    std.append(random_search.cv_results_['std_test_score'])
    best_param=random_search.best_params_
```

Experiment 1

- Iterasi=200
- random state=1

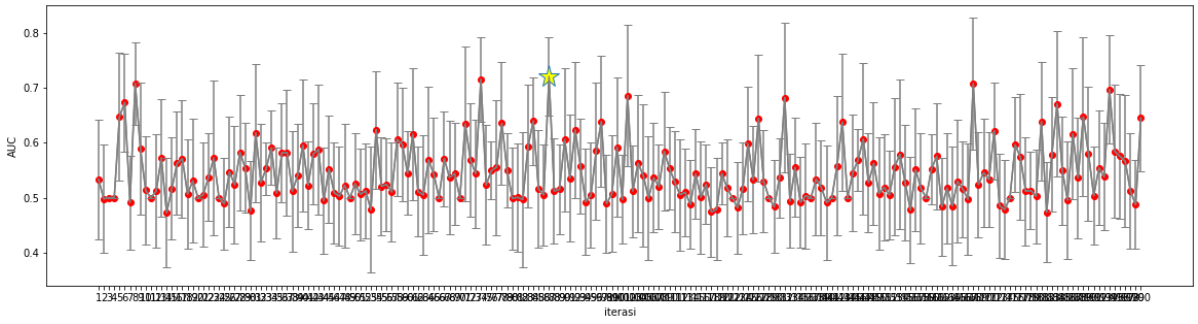
In [6]:
```
iteration=200
auc=[]
std=[]

random_search=RandomizedSearchCV(
              model,
              params2,
              n_iter=iteration,
              scoring='roc_auc',
              n_jobs=-1, cv=5,
              random_state=1)
random_search.fit(X_bank,y_bank)

best_index=random_search.best_index_
auc.append(random_search.cv_results_['mean_test_score'])
std.append(random_search.cv_results_['std_test_score'])
best_param=random_search.best_params_
```

In [7]:
```
print("Iterasi: ",iteration)
print("All AUC: ",auc[0],'\n')
print(np.mean(auc[0]))
print("All std: ",std[0],'\n')
print("Best Hyperparameter: ",best_param,'\n')
print("Best index/iterasi : ",best_index)
print("Best AUC :",auc[0][best_index],"( std:",std[0][best_index],")",'\n')
```

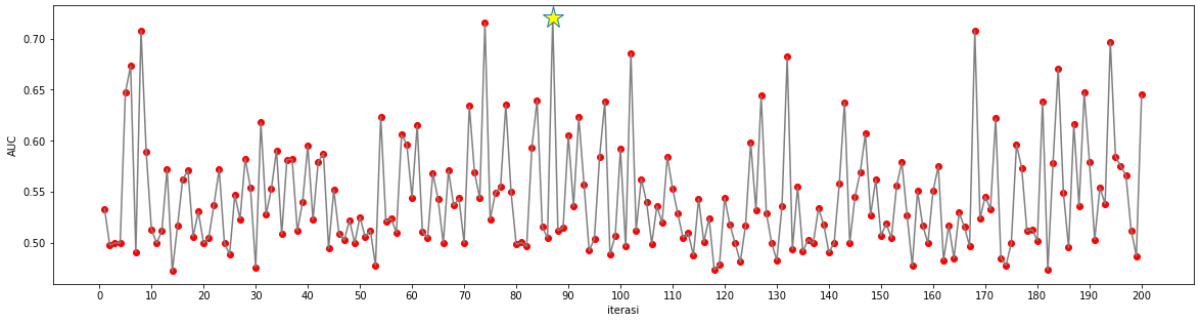
```
Iterasi:  200
All AUC:  [0.53348124 0.49785851 0.5        0.5        0.64739294 0.6731435
 0.49143318 0.70773285 0.58892788 0.51328746 0.5         0.51173498
 0.57243581 0.47250092 0.51666158 0.56237815 0.57097184 0.5063618
 0.53074191 0.5        0.50532118 0.53761297 0.57215295 0.5
 0.48897628 0.54696601 0.52357493 0.58251458 0.55438695 0.47635857
 0.61790796 0.52779491 0.5527723  0.59081176 0.50857256 0.58169562
 0.58205232 0.51170686 0.5399496  0.59530544 0.52262516 0.57954928
 0.58779947 0.49488048 0.55229454 0.50901669 0.50278291 0.52250972
 0.5        0.52552846 0.50615812 0.51245741 0.47793607 0.62381461
 0.52073378 0.52401703 0.51027271 0.60593124 0.59663425 0.54456065
 0.61522565 0.51117102 0.50511311 0.56839844 0.54315057 0.5
 0.57096509 0.53673719 0.54382208 0.5        0.63431218 0.56882867
 0.54369241 0.71545875 0.52309331 0.54916963 0.55542578 0.63533685
 0.55005559 0.49869061 0.50062747 0.49663512 0.59371213 0.63967308
 0.51566973 0.50514839 0.72036085 0.51213222 0.51522888 0.60557275
 0.53571929 0.6232049  0.55726112 0.49253379 0.50439629 0.58463796
 0.63846797 0.48920397 0.50695279 0.59190371 0.49653566 0.68557838
 0.51221953 0.56270605 0.54048358 0.49944633 0.53609028 0.51993061
 0.58448148 0.55340333 0.52878731 0.50546439 0.5096     0.48806865
 0.5434078  0.50081519 0.52453595 0.47394302 0.47895572 0.54420524
 0.51776349 0.5        0.4815376  0.51691493 0.59787901 0.53225126
 0.64410554 0.52945067 0.5        0.48332944 0.53604407 0.68218789
 0.49413363 0.5550185  0.49180991 0.50343397 0.5        0.53390981
 0.51833181 0.49081071 0.5        0.55825061 0.63772859 0.5
 0.5451513  0.56887762 0.60688308 0.52760726 0.56224662 0.50736216
 0.51884321 0.50503066 0.55634494 0.57887115 0.52717886 0.47783647
 0.55083622 0.51714446 0.5        0.55125785 0.57524731 0.48340726
 0.51736399 0.48509502 0.52976497 0.51567547 0.49704708 0.70758506
 0.52407001 0.54544373 0.533061   0.62194196 0.48548371 0.47839226
 0.5        0.59621766 0.57359608 0.51177658 0.5131502  0.50244435
 0.63843126 0.47357082 0.5781774  0.67070166 0.54945649 0.49568402
 0.61613651 0.53639083 0.64705734 0.5794642  0.50354075 0.55428652
 0.53823129 0.69685588 0.58453057 0.57526952 0.56635083 0.51243977
 0.4874274  0.64498743]

0.5447572310215225
All std:  [0.10929969 0.09808629 0.         0.         0.11676919 0.08868502
 0.08529388 0.07450728 0.11988682 0.09822186 0.         0.10295943
 0.10679342 0.0991513  0.09319521 0.09401297 0.10650611 0.09977693
 0.11303552 0.         0.09432527 0.07952898 0.14045341 0.
 0.08317535 0.10095832 0.10747176 0.10496503 0.08238913 0.09039859
 0.1260276  0.10672685 0.04718328 0.06703747 0.08234331 0.09030407
 0.11490687 0.11009411 0.09396971 0.12058881 0.07952849 0.09180459
 0.1187925  0.0967493  0.09784742 0.09148605 0.09044518 0.11250528
 0.         0.09154573 0.08324493 0.08569666 0.1147312  0.10687003
 0.08883511 0.08396378 0.08979044 0.10329093 0.10261953 0.07620044
 0.12032961 0.08173941 0.10832186 0.13322301 0.10334082 0.
 0.08676841 0.10399185 0.09321615 0.         0.14086561 0.10328538
 0.09871244 0.07651684 0.10879194 0.05236358 0.12301839 0.11220622
 0.06692172 0.09261423 0.0982469  0.12348244 0.11182354 0.07979162
 0.10721631 0.09158396 0.07185907 0.09632836 0.08088696 0.13044159
 0.11580772 0.12431985 0.08714445 0.09750125 0.09576199 0.12519105
 0.11936783 0.08902269 0.10623799 0.12624642 0.08071832 0.12889513
 0.08358865 0.12453248 0.1300323  0.11222581 0.09931707 0.07745766
 0.10798511 0.07569076 0.09175096 0.10115844 0.08049888 0.08005746
 0.082461   0.1024338  0.07178428 0.08177185 0.09304377 0.05679759
```

```
 0.08846015 0.         0.08392977 0.08322557 0.10339147 0.09774966
 0.11510238 0.09298489 0.         0.08454762 0.07260783 0.13570425
 0.08515228 0.08974789 0.08206251 0.09585841 0.         0.10263373
 0.08635342 0.10376268 0.         0.12637524 0.12480721 0.
 0.10612181 0.056662   0.13759659 0.09046051 0.11115563 0.10225686
 0.09611559 0.08072129 0.12488491 0.13605162 0.10483281 0.09704402
 0.11198263 0.10010623 0.         0.06363705 0.09621165 0.08949732
 0.10086536 0.10833375 0.11148525 0.08701586 0.10182292 0.11944756
 0.09534248 0.09911793 0.08679485 0.08805132 0.094797   0.09035092
 0.         0.0861153  0.1145713  0.09875569 0.0706163  0.08542738
 0.10786514 0.09072099 0.1042639  0.13188845 0.09836704 0.10658383
 0.11933281 0.10962013 0.14494977 0.12246822 0.08895589 0.10460761
 0.09729067 0.0990915  0.12114987 0.11320942 0.12002646 0.10440358
 0.08052384 0.0964038 ]

Best Hyperparameter:  {'subsample': 0.05263157894736842, 'min_child_weight':
14, 'max_depth': 45, 'gamma': 0.9183673469387754, 'eta': 0.8183636363636364}

Best index/iterasi :  86
Best AUC : 0.7203608516313473 ( std: 0.07185906796685554 )
```

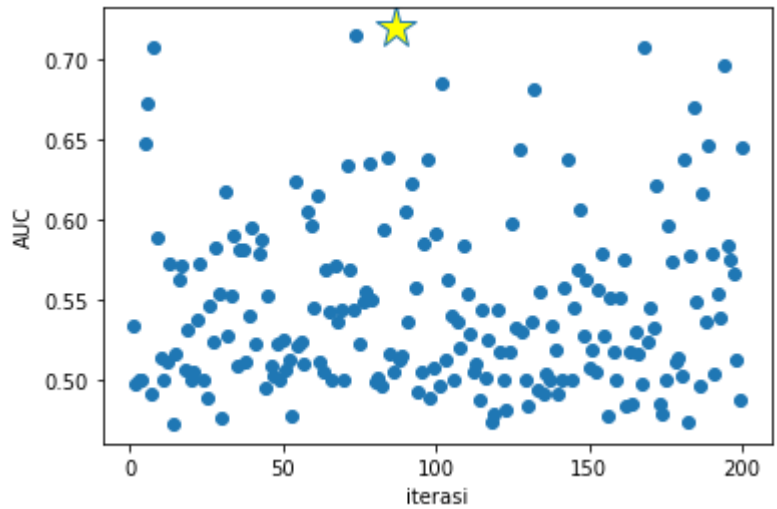
In [23]:
```
import matplotlib.pyplot as plt
fig=plt.figure(figsize=(20,5))
x=range(1,iteration+1)
y=auc[0]
error=std[0]
plt.plot(x,y,'or')
plt.plot(x,y,color='gray')
plt.plot(x[best_index],y[best_index],marker='*',markersize=22,markerfacecolor=
'yellow')
plt.errorbar(x, y, yerr=error,color='gray',capsize=4)
fig=plt.xticks(range(1,iteration+1))
fig=plt.xlabel('iterasi')
fig=plt.ylabel('AUC')
```



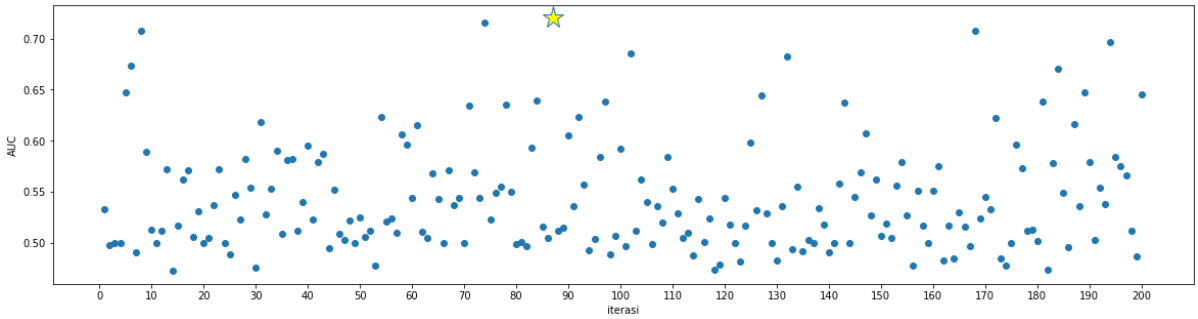
In [21]:
```
fig=plt.figure(figsize=(20,5))
x=range(1,iteration+1)
y=auc[0]
error=std[0]
plt.plot(x,y,'or')
plt.plot(x,y,color='gray')
plt.plot(x[best_index],y[best_index],marker='*',markersize=22,markerfacecolor=
'yellow')
fig=plt.xticks(range(1,iteration+1))
fig=plt.xlabel('iterasi')
fig=plt.ylabel('AUC')
fig=plt.xticks(range(0,210,10))
```



In [17]:
```
fig=plt.figure()
x=range(1,iteration+1)
y=auc[0]
error=std[0]
plt.scatter(x,y)
plt.plot(x[best_index],y[best_index],marker='*',markersize=22,markerfacecolor=
'yellow')
fig=plt.xticks(range(1,iteration+1))
fig=plt.xlabel('iterasi')
fig=plt.ylabel('AUC')
fig=plt.xticks(range(0,210,50))
```



In [22]:
```
fig=plt.figure(figsize=(20,5))
x=range(1,iteration+1)
y=auc[0]
error=std[0]
plt.scatter(x,y)
plt.plot(x[best_index],y[best_index],marker='*',markersize=22,markerfacecolor=
'yellow')
fig=plt.xticks(range(1,iteration+1))
fig=plt.xlabel('iterasi')
fig=plt.ylabel('AUC')
fig=plt.xticks(range(0,210,10))
```



Experiment 2

- random search, random state = 42

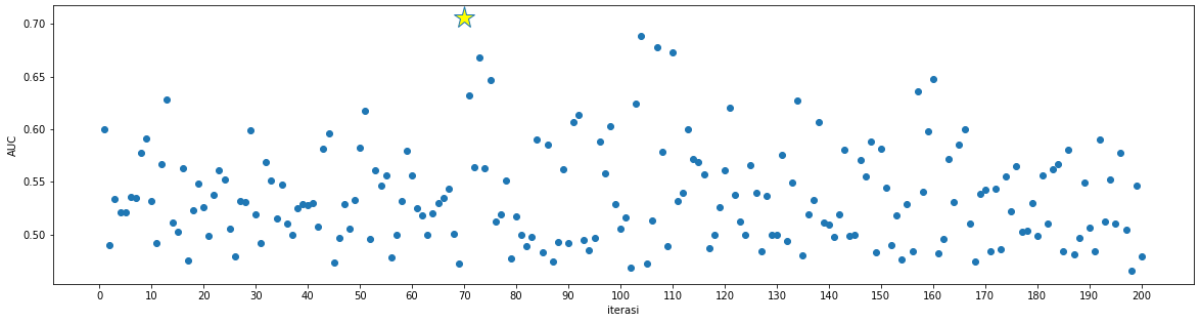
In [5]:
```
iteration=200
auc=[]
std=[]

random_search=RandomizedSearchCV(
                model,
                params2,
                n_iter=iteration,
                scoring='roc_auc',
                n_jobs=-1, cv=5,
                random_state=42)
random_search.fit(X_bank,y_bank)

best_index=random_search.best_index_
auc.append(random_search.cv_results_['mean_test_score'])
std.append(random_search.cv_results_['std_test_score'])
best_param=random_search.best_params_
```

In [6]:
```
print("Iterasi: ",iteration)
print("All AUC: ",auc[0],'\n')
print(np.mean(auc[0]))
print("All std: ",std[0],'\n')
print("Best Hyperparameter: ",best_param,'\n')
print("Best index/iterasi : ",best_index)
print("Best AUC :",auc[0][best_index],"( std:",std[0][best_index],")",'\n')
```

```
Iterasi:  200
All AUC:  [0.59965364 0.48944121 0.53350531 0.52119794 0.5209616  0.53584633
 0.53474915 0.57718286 0.59098183 0.53136263 0.49165171 0.56640098
 0.62846264 0.51155754 0.50238166 0.56301111 0.4750668  0.52307407
 0.54861022 0.52544539 0.49898764 0.53720633 0.56111093 0.55224083
 0.50553189 0.47934403 0.53170529 0.53108781 0.59880321 0.51877683
 0.4916416  0.5689489  0.55159403 0.51490018 0.54736678 0.5103221
 0.49933828 0.5247645  0.52884025 0.52782506 0.52986884 0.5071358
 0.58108133 0.59612775 0.47331097 0.49672272 0.52907999 0.50512677
 0.53316183 0.58269584 0.61730638 0.49537314 0.56112582 0.54676051
 0.55601106 0.47793256 0.5        0.5318224  0.57972197 0.55619497
 0.52471717 0.518066   0.5        0.52031292 0.53014372 0.53438318
 0.5431883  0.50019558 0.47192994 0.70577529 0.63165506 0.56423113
 0.66815463 0.56335795 0.64639909 0.51183592 0.51918778 0.55120595
 0.47689401 0.51699376 0.5        0.48864616 0.49775843 0.59046167
 0.4825306  0.58564022 0.47410663 0.49291521 0.5614371  0.49160823
 0.60666412 0.61397324 0.49461875 0.48455391 0.49648987 0.5878868
 0.55808838 0.60302981 0.52845044 0.50528987 0.51659569 0.46821669
 0.6242343  0.68872597 0.4722058  0.51281694 0.67782457 0.57885569
 0.48908619 0.67347488 0.53157552 0.53976338 0.60010233 0.571438
 0.56841439 0.55695968 0.48667541 0.5        0.52543449 0.56076654
 0.62008434 0.5379395  0.5121998  0.5        0.56573497 0.53947877
 0.48413733 0.53656104 0.5        0.5        0.57545968 0.4940535
 0.54905789 0.62690795 0.48039429 0.51944072 0.53240355 0.60648815
 0.51134536 0.50902794 0.49770334 0.51863857 0.58001444 0.49900522
 0.49921573 0.57063442 0.55462791 0.58833074 0.48316853 0.58114372
 0.54439202 0.49017045 0.51819981 0.47613365 0.52899403 0.48439747
 0.63580228 0.54001977 0.59771413 0.64726435 0.48219239 0.49612002
 0.57138429 0.53108175 0.58492165 0.59992495 0.51044989 0.47463897
 0.53873644 0.54280464 0.48425712 0.54369138 0.48626577 0.55471959
 0.52165948 0.56453664 0.50243929 0.50298543 0.52971065 0.49906856
 0.55583796 0.51055169 0.56207538 0.56684639 0.48401467 0.58029506
 0.48079509 0.4962749  0.54952975 0.50679266 0.4841785  0.59007855
 0.51270979 0.55233461 0.50984921 0.57766051 0.50470014 0.46502541
 0.5459903  0.47918775]

0.5373299381722112
All std:  [0.10880914 0.09041341 0.10369243 0.10192557 0.09241942 0.08593435
 0.10422915 0.1312237  0.1020159  0.10238402 0.11255094 0.12356707
 0.11690244 0.10212985 0.09826907 0.08273013 0.08409163 0.10281072
 0.10861872 0.08078655 0.12591291 0.10234932 0.0976186  0.07014175
 0.08560033 0.09574229 0.09697091 0.11247188 0.0920979  0.08730767
 0.08462017 0.10277466 0.1027017  0.09269894 0.08936689 0.11409365
 0.10801399 0.09210102 0.09527237 0.09303056 0.0976291  0.09882224
 0.09389996 0.09814228 0.09419906 0.09341855 0.12292904 0.10420148
 0.08955676 0.09502618 0.11331195 0.07150086 0.10817658 0.11103502
 0.12869123 0.08976445 0.         0.10341295 0.09046244 0.11106831
 0.07685804 0.10799911 0.         0.11728454 0.10445092 0.07541622
 0.08557719 0.09252215 0.08853595 0.07318935 0.11692992 0.08354017
 0.12179925 0.07017542 0.13252775 0.09100185 0.09116429 0.12874452
 0.08744515 0.08408714 0.         0.08743806 0.08144497 0.10887688
 0.10150156 0.08697787 0.09793042 0.11003526 0.10070787 0.0887812
 0.06820162 0.09648463 0.0811446  0.08564762 0.09079481 0.1105662
 0.07193962 0.10290036 0.09862563 0.0836293  0.08620794 0.1022663
 0.1263836  0.13102906 0.09608235 0.07020083 0.12945353 0.09960442
 0.10968778 0.13200327 0.06926509 0.12743327 0.12331852 0.04987313
 0.09635396 0.08973374 0.10618286 0.         0.10814763 0.09650926
```

```
 0.11314389 0.09260194 0.09274566 0.         0.08190394 0.1157667
 0.07621761 0.09302197 0.         0.         0.12744341 0.082129
 0.06730649 0.12833874 0.10727054 0.09171583 0.12186282 0.0822854
 0.10225341 0.06809773 0.1044079  0.1115064  0.11553146 0.10013381
 0.07604989 0.1323187  0.09425241 0.12446412 0.0903668  0.08512888
 0.09899594 0.1110124  0.11155918 0.09363142 0.09185173 0.10050932
 0.09487566 0.13361271 0.12485624 0.1371171  0.09053747 0.09913647
 0.11201099 0.08794564 0.10125866 0.11835755 0.09607545 0.09260733
 0.09015033 0.09003173 0.10012232 0.11923115 0.05090813 0.12421615
 0.10143229 0.11672363 0.08800677 0.09747382 0.11429527 0.10088833
 0.11524541 0.09204588 0.09704518 0.12852676 0.06960786 0.1199551
 0.08894065 0.08743203 0.09314648 0.09508196 0.10043198 0.11763653
 0.10720593 0.05702549 0.07893805 0.09563106 0.09987126 0.10678937
 0.09089117 0.08671596]

Best Hyperparameter:  {'subsample': 0.05263157894736842, 'min_child_weight':
12, 'max_depth': 15, 'gamma': 0.5918367346938775, 'eta': 1.0}

Best index/iterasi :  69
Best AUC : 0.7057752856162385 ( std: 0.07318934568303126 )
```

In [8]:
```
import matplotlib.pyplot as plt
fig=plt.figure(figsize=(20,5))
x=range(1,iteration+1)
y=auc[0]
error=std[0]
plt.scatter(x,y)
plt.plot(x[best_index],y[best_index],marker='*',markersize=22,markerfacecolor=
'yellow')
fig=plt.xticks(range(1,iteration+1))
fig=plt.xlabel('iterasi')
fig=plt.ylabel('AUC')
fig=plt.xticks(range(0,210,10))
```



Experiment 3

In [ ]:
```

```