In [1]:
```python
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib as mpl
import matplotlib.pyplot as plt
```

In [2]:
```python
import tensorflow as tf
from tensorflow import keras
```

In [3]:
```python
from sklearn.datasets import fetch_california_housing
housing = fetch_california_housing()
```

In [4]:
```python
print(housing.feature_names)
```

```
['MedInc', 'HouseAge', 'AveRooms', 'AveBedrms', 'Population', 'AveOccup', 'Latitude', 'Longitude']
```

In [5]:
```python
from sklearn.model_selection import train_test_split
X_train_full, X_test, y_train_full, y_test = train_test_split(housing.data, housing.target, random_state = 42)
X_train, X_valid, y_train, y_valid = train_test_split(X_train_full, y_train_full, random_state = 42)
```

In [6]:
```python
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_valid = scaler.transform(X_valid)
X_test = scaler.transform(X_test)
```

In [7]:
```python
np.random.seed(42)
tf.random.set_seed(42)
```

In [8]:
```python
X_train.shape
```

Out[8]:
```
(11610, 8)
```

In [9]:
```python
model = keras.models.Sequential([
    keras.layers.Dense(30, activation = "relu", input_shape=[8]),
    keras.layers.Dense(30, activation = "relu"),
    keras.layers.Dense(1)
])
```

In [10]:
```python
model.compile(loss = "mean_squared_error",
              optimizer = keras.optimizers.SGD(lr=1e-3),
              metrics = ["mae"])
```

In [11]:
```python
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense (Dense)                (None, 30)                270       
_________________________________________________________________
dense_1 (Dense)              (None, 30)                930       
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 31        
=================================================================
Total params: 1,231
Trainable params: 1,231
Non-trainable params: 0
_________________________________________________________________
```

In [12]:
```python
model_history = model.fit(X_train, y_train, epochs = 20, validation_data = (X_valid, y_valid))
```

```
Epoch 1/20
363/363 [==============================] - 1s 1ms/step - loss: 1.8866 - mae: 0.9900 - val_loss: 0.7126 - val_mae: 0.6368
Epoch 2/20
363/363 [==============================] - 0s 1ms/step - loss: 0.6577 - mae: 0.6042 - val_loss: 0.6880 - val_mae: 0.5704
Epoch 3/20
363/363 [==============================] - 0s 1ms/step - loss: 0.5934 - mae: 0.5618 - val_loss: 0.5803 - val_mae: 0.5352
Epoch 4/20
363/363 [==============================] - 0s 1ms/step - loss: 0.5557 - mae: 0.5398 - val_loss: 0.5166 - val_mae: 0.5207
Epoch 5/20
363/363 [==============================] - 0s 1ms/step - loss: 0.5272 - mae: 0.5237 - val_loss: 0.4895 - val_mae: 0.5022
Epoch 6/20
363/363 [==============================] - 0s 1ms/step - loss: 0.5033 - mae: 0.5113 - val_loss: 0.4951 - val_mae: 0.4934
Epoch 7/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4854 - mae: 0.5010 - val_loss: 0.4861 - val_mae: 0.4838
Epoch 8/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4709 - mae: 0.4924 - val_loss: 0.4554 - val_mae: 0.4753
Epoch 9/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4578 - mae: 0.4857 - val_loss: 0.4413 - val_mae: 0.4671
Epoch 10/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4474 - mae: 0.4797 - val_loss: 0.4379 - val_mae: 0.4623
Epoch 11/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4393 - mae: 0.4744 - val_loss: 0.4396 - val_mae: 0.4638
Epoch 12/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4318 - mae: 0.4703 - val_loss: 0.4507 - val_mae: 0.4573
Epoch 13/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4261 - mae: 0.4674 - val_loss: 0.3997 - val_mae: 0.4517
Epoch 14/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4202 - mae: 0.4636 - val_loss: 0.3956 - val_mae: 0.4497
Epoch 15/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4155 - mae: 0.4613 - val_loss: 0.3916 - val_mae: 0.4464
Epoch 16/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4112 - mae: 0.4591 - val_loss: 0.3937 - val_mae: 0.4445
Epoch 17/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4077 - mae: 0.4569 - val_loss: 0.3809 - val_mae: 0.4390
Epoch 18/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4040 - mae: 0.4545 - val_loss: 0.3793 - val_mae: 0.4368
Epoch 19/20
363/363 [==============================] - 0s 1ms/step - loss: 0.4004 - mae: 0.4521 - val_loss: 0.3850 - val_mae: 0.4369
Epoch 20/20
363/363 [==============================] - 0s 1ms/step - loss: 0.3980 - mae: 0.4508 - val_loss: 0.3809 - val_mae: 0.4368
```

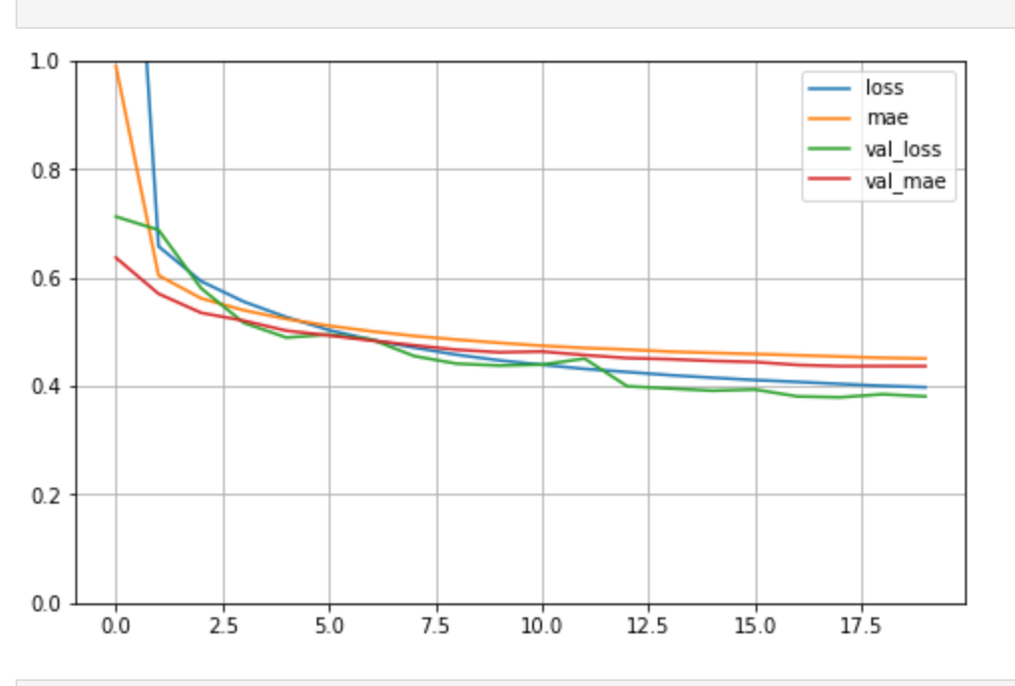
In [13]:
```python
mae_test = model.evaluate(X_test, y_test)
```

```
162/162 [==============================] - 0s 808us/step - loss: 0.3942 - mae: 0.4502
```

In [14]:
```python
model_history.history
```

Out[14]:
```
{'loss': [1.8866397142410278,
  0.6577126979827881,
  0.593418538570404,
  0.5557191371917725,
  0.5271904468536377,
  0.5032975673675537,
  0.48535552620887756,
  0.47091808915138245,
  0.45779937505722046,
  0.4474469721317291,
  0.4393136501312256,
  0.43176087737083435,
  0.42605164647102356,
  0.42017653584480286,
  0.41549986600875854,
  0.4111650288105011,
  0.4077068865299225,
  0.40395283699035645,
  0.4004494547843933,
  0.39796024560928345],
 'mae': [0.9900256991386414,
  0.6041509509086609,
  0.5618006587028503,
  0.5398454070091248,
  0.5237293839454651,
  0.5112563371658325,
  0.5010154247283936,
  0.492448091506958,
  0.4857262969017029,
  0.4797375202178955,
  0.4744163453578949,
  0.4703480303287506,
  0.46740883588790894,
  0.46360209584236145,
  0.461266428232193,
  0.4591343402862549,
  0.45687004923820496,
  0.4545365273952484,
  0.4521065056324005,
  0.45083147287368774],
 'val_loss': [0.7126054763793945,
  0.6880088448524475,
  0.580328643321991,
  0.5166085362434387,
  0.48950764536857605,
  0.4950792193412781,
  0.4861253798007965,
  0.4553801715373993,
  0.44133713841438293,
  0.4378637969493866,
  0.43964388966560364,
  0.4506688714027405,
  0.39972347021102905,
  0.39558935165405273,
  0.391572505235672,
  0.3936831057071686,
  0.3809485137462616,
  0.3793475329875946,
  0.3850175738334656,
  0.380946546792984],
 'val_mae': [0.6368111968040466,
  0.5703656673431396,
  0.5351505279541016,
  0.5206614136695862,
  0.5022227168083191,
  0.4933752417564392,
  0.48384901881217957,
  0.47527745366096497,
  0.46705934405326843,
  0.46234598755836487,
  0.46377918124198914,
  0.4572649896144867,
  0.45166537165641785,
  0.4496610760688782,
  0.4463699758052826,
  0.4444962739944458,
  0.43897417187690735,
  0.43681108951568604,
  0.4369049668312073,
  0.43676161766052246]}
```

In [15]:
```python
import pandas as pd
pd.DataFrame(model_history.history).plot(figsize = (8,5))
plt.grid(True)
plt.gca().set_ylim(0,1)
plt.show()
```



In [16]:
```python
X_new = X_test[:3]
```

In [17]:
```python
y_pred = model.predict(X_new)
print(y_pred)
print(y_test[:3])
```

```
[[0.5328768]
 [1.8915398]
 [3.404087 ]]
[0.477   0.458   5.00001]
```

Functional API

In [18]:
```python
del model
```

In [19]:
```python
keras.backend.clear_session()
```

In [20]:
```python
input_ = keras.layers.Input(shape =X_train.shape[1:])
hidden1 = keras.layers.Dense(30, activation = "relu")(input_)
hidden2 = keras.layers.Dense(30, activation = "relu")(hidden1)
concat = keras.layers.concatenate([input_, hidden2])
output = keras.layers.Dense(1)(concat)
model = keras.models.Model(inputs = [input_], outputs = [output])
```

In [21]:
```python
model.summary()
```

```
Model: "functional_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to                     
==================================================================================================
input_1 (InputLayer)            [(None, 8)]          0                                            
__________________________________________________________________________________________________
dense (Dense)                   (None, 30)           270         input_1[0][0]                    
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 30)           930         dense[0][0]                      
__________________________________________________________________________________________________
concatenate (Concatenate)       (None, 38)           0           input_1[0][0]                    
                                                                 dense_1[0][0]                    
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 1)            39          concatenate[0][0]                
==================================================================================================
Total params: 1,239
Trainable params: 1,239
Non-trainable params: 0
__________________________________________________________________________________________________
```

In [22]:
```python
model.compile(loss = "mean_squared_error",
              optimizer = keras.optimizers.SGD(lr=1e-3),
              metrics = ["mae"])
```

In [28]:
```python
model_history = model.fit(X_train, y_train, epochs = 50, validation_data = (X_valid, y_valid))
```

```
Epoch 1/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3710 - mae: 0.4339 - val_loss: 0.3721 - val_mae: 0.4246
Epoch 2/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3700 - mae: 0.4326 - val_loss: 0.4372 - val_mae: 0.4316
Epoch 3/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3692 - mae: 0.4328 - val_loss: 0.4056 - val_mae: 0.4266
Epoch 4/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3680 - mae: 0.4315 - val_loss: 0.3777 - val_mae: 0.4268
Epoch 5/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3675 - mae: 0.4310 - val_loss: 0.3728 - val_mae: 0.4246
Epoch 6/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3663 - mae: 0.4304 - val_loss: 0.3962 - val_mae: 0.4245
Epoch 7/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3658 - mae: 0.4297 - val_loss: 0.3729 - val_mae: 0.4236
Epoch 8/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3649 - mae: 0.4294 - val_loss: 0.3579 - val_mae: 0.4204
Epoch 9/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3640 - mae: 0.4287 - val_loss: 0.3884 - val_mae: 0.4221
Epoch 10/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3633 - mae: 0.4281 - val_loss: 0.3942 - val_mae: 0.4218
Epoch 11/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3627 - mae: 0.4273 - val_loss: 0.3469 - val_mae: 0.4183
Epoch 12/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3617 - mae: 0.4268 - val_loss: 0.4210 - val_mae: 0.4234
Epoch 13/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3612 - mae: 0.4264 - val_loss: 0.3765 - val_mae: 0.4222
Epoch 14/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3604 - mae: 0.4255 - val_loss: 0.3525 - val_mae: 0.4193
Epoch 15/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3597 - mae: 0.4254 - val_loss: 0.3837 - val_mae: 0.4212
Epoch 16/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3590 - mae: 0.4251 - val_loss: 0.3434 - val_mae: 0.4169
Epoch 17/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3583 - mae: 0.4244 - val_loss: 0.3974 - val_mae: 0.4204
Epoch 18/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3581 - mae: 0.4242 - val_loss: 0.3741 - val_mae: 0.4175
Epoch 19/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3571 - mae: 0.4233 - val_loss: 0.3637 - val_mae: 0.4171
Epoch 20/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3565 - mae: 0.4229 - val_loss: 0.4175 - val_mae: 0.4211
Epoch 21/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3560 - mae: 0.4223 - val_loss: 0.3743 - val_mae: 0.4196
Epoch 22/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3552 - mae: 0.4224 - val_loss: 0.4306 - val_mae: 0.4226
Epoch 23/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3548 - mae: 0.4223 - val_loss: 0.3367 - val_mae: 0.4109
Epoch 24/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3540 - mae: 0.4208 - val_loss: 0.3773 - val_mae: 0.4162
Epoch 25/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3534 - mae: 0.4209 - val_loss: 0.3935 - val_mae: 0.4174
Epoch 26/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3530 - mae: 0.4205 - val_loss: 0.3899 - val_mae: 0.4164
Epoch 27/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3527 - mae: 0.4203 - val_loss: 0.3367 - val_mae: 0.4111
Epoch 28/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3520 - mae: 0.4194 - val_loss: 0.3803 - val_mae: 0.4172
Epoch 29/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3516 - mae: 0.4196 - val_loss: 0.3591 - val_mae: 0.4147
Epoch 30/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3508 - mae: 0.4189 - val_loss: 0.3871 - val_mae: 0.4146
Epoch 31/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3501 - mae: 0.4182 - val_loss: 0.3525 - val_mae: 0.4119
Epoch 32/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3497 - mae: 0.4180 - val_loss: 0.3920 - val_mae: 0.4172
Epoch 33/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3496 - mae: 0.4184 - val_loss: 0.3446 - val_mae: 0.4118
Epoch 34/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3487 - mae: 0.4175 - val_loss: 0.3494 - val_mae: 0.4126
Epoch 35/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3484 - mae: 0.4172 - val_loss: 0.3984 - val_mae: 0.4167
Epoch 36/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3478 - mae: 0.4172 - val_loss: 0.3323 - val_mae: 0.4083
Epoch 37/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3475 - mae: 0.4163 - val_loss: 0.3625 - val_mae: 0.4125
Epoch 38/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3470 - mae: 0.4165 - val_loss: 0.3396 - val_mae: 0.4094
Epoch 39/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3463 - mae: 0.4157 - val_loss: 0.3447 - val_mae: 0.4092
Epoch 40/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3457 - mae: 0.4155 - val_loss: 0.3507 - val_mae: 0.4081
Epoch 41/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3458 - mae: 0.4150 - val_loss: 0.3616 - val_mae: 0.4102
Epoch 42/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3455 - mae: 0.4153 - val_loss: 0.3574 - val_mae: 0.4101
Epoch 43/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3446 - mae: 0.4142 - val_loss: 0.3601 - val_mae: 0.4097
Epoch 44/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3440 - mae: 0.4140 - val_loss: 0.3734 - val_mae: 0.4126
Epoch 45/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3441 - mae: 0.4144 - val_loss: 0.3374 - val_mae: 0.4060
Epoch 46/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3431 - mae: 0.4129 - val_loss: 0.3473 - val_mae: 0.4084
Epoch 47/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3429 - mae: 0.4135 - val_loss: 0.3490 - val_mae: 0.4072
Epoch 48/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3424 - mae: 0.4127 - val_loss: 0.3625 - val_mae: 0.4098
Epoch 49/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3427 - mae: 0.4130 - val_loss: 0.3320 - val_mae: 0.4068
Epoch 50/50
363/363 [==============================] - 1s 2ms/step - loss: 0.3416 - mae: 0.4126 - val_loss: 0.3254 - val_mae: 0.4038
```

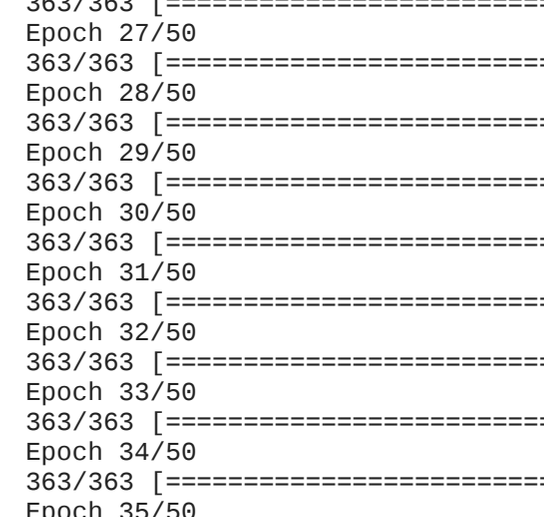
In [29]:
```python
mae_test = model.evaluate(X_test, y_test)
```

```
162/162 [==============================] - 0s 1ms/step - loss: 0.3421 - mae: 0.4117
```

In [30]:
```python
model_history.history
```

Out[30]:
```
{'loss': [0.3709588646888733,
  0.3700377941131592,
  0.3691806495189667,
  0.36803939938545227,
  0.3675456941127777,
  0.3662607669830322,
  0.36584290862083435,
  0.3648802638053894,
  0.3639710545539856,
  0.363277792930603,
  0.36274388432502747,
  0.36174845695495605,
  0.3612297475337982,
  0.36037981510162354,
  0.3596731722354889,
  0.35896065831184387,
  0.35828879475593567,
  0.3581177890300751,
  0.3571260869503021,
  0.35652226209640503,
  0.35601919889450073,
  0.3551953434944153,
  0.3548322319984436,
  0.3540223240852356,
  0.3534010648727417,
  0.35303831100463867,
  0.35274970531463623,
  0.35195890069007874,
  0.3515706956386566,
  0.35076096653938293,
  0.35005488991737366,
  0.3497293293476105,
  0.3495754897594452,
  0.3487187922000885,
  0.34836649894714355,
  0.3477964997291565,
  0.34748774766921997,
  0.34695759415626526,
  0.34626224637031555,
  0.3456905782222748,
  0.3457893133163452,
  0.34546419978141785,
  0.3446158170700073,
  0.34396031498908997,
  0.3441402316093445,
  0.34314095973968506,
  0.34288734197616577,
  0.34237080812454224,
  0.34265634417533875,
  0.3415549397468567],
 'mae': [0.43392378091812134,
  0.4326052963733673,
  0.4328071177005768,
  0.4315342903137207,
  0.43099021911621094,
  0.43039077520370483,
  0.4296959936618805,
  0.42937299609184265,
  0.4287203252315521,
  0.42812028527259827,
  0.4273405075073242,
  0.42676690220832825,
  0.4264203608036041,
  0.42547348141670227,
  0.4254412055015564,
  0.4250515103340149,
  0.42441755533218384,
  0.42419376969337463,
  0.4232858121395111,
  0.4228838086128235,
  0.4223436117172241,
  0.42237189412117004,
  0.4222821891307831,
  0.4208475947380066,
  0.42085182666778564,
  0.42051246762275696,
  0.4202641248703003,
  0.4194088280200958,
  0.41958239674568176,
  0.4189318120479584,
  0.4182455837726593,
  0.41795453429222107,
  0.41836270689964294,
  0.4175094962120056,
  0.4172460436820984,
  0.4171817898750305,
  0.41634127497673035,
  0.4164685904979706,
  0.4156774580478668,
  0.4155293107032776,
  0.41498932242393494,
  0.41525211930274963,
  0.4141796827316284,
  0.4139708876609802,
  0.4144400954246521,
  0.4129040837287903,
  0.4135182499885559,
  0.4126714766025543,
  0.4130064845085144,
  0.41255080699920654],
 'val_loss': [0.3720780611038208,
  0.43724170327186584,
  0.4055604040622711,
  0.3776942491531372,
  0.37283656001091003,
  0.39617401361465454,
  0.3728782832622528,
  0.35790398716926575,
  0.3883959650993347,
  0.3942481279373169,
  0.3468644917011261,
  0.42103755474090576,
  0.37646907567977905,
  0.3525142967700958,
  0.3836924731731415,
  0.343424528837204,
  0.39742767810821533,
  0.37407878041267395,
  0.36368894577026367,
  0.41746413707733154,
  0.37429073452949524,
  0.4306071400642395,
  0.3366701304912567,
  0.3773202896118164,
  0.3935299515724182,
  0.38991647958755493,
  0.33670052886009216,
  0.3803355097770691,
  0.3590599298477173,
  0.38706204295158386,
  0.3525407612323761,
  0.39203229546546936,
  0.3446025848388672,
  0.3494240641593933,
  0.39840009808540344,
  0.33234959840774536,
  0.3625227212905884,
  0.3395546078681946,
  0.34473633766174316,
  0.350690096616745,
  0.3615897297859192,
  0.3573894798755646,
  0.3601314425468445,
  0.37339362502098083,
  0.33744746446609497,
  0.3472752273082733,
  0.3490368127822876,
  0.3625413179397583,
  0.3320060670375824,
  0.3254346549510956],
 'val_mae': [0.42461284995079040,
  0.4315718710422516,
  0.42656299471855164,
  0.4268057644367218,
  0.4246273934841156,
  0.42445632815361023,
  0.42360982298851013,
  0.4203636944293976,
  0.42214080691337585,
  0.4218372106552124,
  0.41829368472099304,
  0.4233700931072235,
  0.42216238379478455,
  0.41930705308914185,
  0.42117342352867126,
  0.4168858826160431,
  0.4203789532184601,
  0.4174744486808777,
  0.41714194416999817,
  0.4211013913154602,
  0.4196149706840515,
  0.42261195182800293,
  0.4108779728412628,
  0.4161524772644043,
  0.4173585772514343,
  0.4164314270019531,
  0.4110642671585083,
  0.4171706736087799,
  0.4146653711795807,
  0.4146343767642975,
  0.41189369559288025,
  0.41719362139701843,
  0.41183099150657654,
  0.41257086396217346,
  0.4167420268058777,
  0.4083348214626312,
  0.4125193953514099,
  0.40942689776420593,
  0.4092250466346741,
  0.4080788493156433,
  0.4102322459220886,
  0.4100853502750397,
  0.4097105860710144,
  0.4125751554965973,
  0.40597543120384216,
  0.4083980917930603,
  0.407164990901947,
  0.409760445356369,
  0.40682294964790344,
  0.4037694036960602]}
```

In [33]:
```python
import pandas as pd
pd.DataFrame(model_history.history).plot(figsize = (4,3))
plt.grid(True)
plt.gca().set_ylim(0,1)
plt.show()
```



In [34]:
```python
model.save("my_func_model.h5")
```

In [35]:
```python
%pwd
```

Out[35]:
```
'C:\\Users\\h22de'
```

In [36]:
```python
del model
```

In [38]:
```python
keras.backend.clear_session()
```

In [39]:
```python
model = keras.models.load_model("my_func_model.h5")
```

In [40]:
```python
model.summary()
```

```
Model: "functional_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to                     
==================================================================================================
input_1 (InputLayer)            [(None, 8)]          0                                            
__________________________________________________________________________________________________
dense (Dense)                   (None, 30)           270         input_1[0][0]                    
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 30)           930         dense[0][0]                      
__________________________________________________________________________________________________
concatenate (Concatenate)       (None, 38)           0           input_1[0][0]                    
                                                                 dense_1[0][0]                    
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 1)            39          concatenate[0][0]                
==================================================================================================
Total params: 1,239
Trainable params: 1,239
Non-trainable params: 0
__________________________________________________________________________________________________
```

In [41]:
```python
y_pred = model.predict(X_new)
print(y_pred)
```

```
[[0.70474243]
 [1.6826377 ]
 [4.186458  ]]
```

Using CallBacks during Training

In [42]:
```python
del model
```

In [43]:
```python
keras.backend.clear_session()
np.random.seed(42)
tf.random.set_seed(42)
```

In [44]:
```python
model = keras.models.Sequential([
    keras.layers.Dense(30, activation = "relu", input_shape=[8]),
    keras.layers.Dense(30, activation = "relu"),
    keras.layers.Dense(1)
])
```

In [45]:
```python
model.compile(loss = "mean_squared_error",
              optimizer = keras.optimizers.SGD(lr=1e-3),
              metrics = ["mae"])
```

In [46]:
```python
checkpoint_cb = keras.callbacks.ModelCheckpoint("Model-{epoch:02d}.h5")
```

In [47]:
```python
history = model.fit(X_train, y_train, epochs = 10, validation_data =(X_valid, y_valid), callbacks = [checkpoint_cb])
```

```
Epoch 1/10
363/363 [==============================] - 0s 951us/step - loss: 1.8866 - mae: 0.9900 - val_loss: 0.7126 - val_mae: 0.6368
Epoch 2/10
363/363 [==============================] - 0s 657us/step - loss: 0.6577 - mae: 0.6042 - val_loss: 0.6880 - val_mae: 0.5704
Epoch 3/10
363/363 [==============================] - 0s 681us/step - loss: 0.5934 - mae: 0.5618 - val_loss: 0.5803 - val_mae: 0.5352
Epoch 4/10
363/363 [==============================] - 0s 667us/step - loss: 0.5557 - mae: 0.5398 - val_loss: 0.5166 - val_mae: 0.5207
Epoch 5/10
363/363 [==============================] - 0s 848us/step - loss: 0.5272 - mae: 0.5237 - val_loss: 0.4895 - val_mae: 0.5022
Epoch 6/10
363/363 [==============================] - 0s 858us/step - loss: 0.5033 - mae: 0.5113 - val_loss: 0.4951 - val_mae: 0.4934
Epoch 7/10
363/363 [==============================] - 0s 821us/step - loss: 0.4854 - mae: 0.5010 - val_loss: 0.4861 - val_mae: 0.4838
Epoch 8/10
363/363 [==============================] - 0s 679us/step - loss: 0.4709 - mae: 0.4924 - val_loss: 0.4554 - val_mae: 0.4753
Epoch 9/10
363/363 [==============================] - 0s 693us/step - loss: 0.4578 - mae: 0.4857 - val_loss: 0.4413 - val_mae: 0.4671
Epoch 10/10
363/363 [==============================] - 0s 731us/step - loss: 0.4474 - mae: 0.4797 - val_loss: 0.4379 - val_mae: 0.4623
```

In [48]:
```python
del  model
keras.backend.clear_session()
```

In [49]:
```python
model = keras.models.load_model("Model-10.h5")
```

In [50]:
```python
mse_test =model.evaluate(X_test, y_test)
```

```
162/162 [==============================] - 0s 442us/step - loss: 0.4382 - mean_absolute_error: 0.4727
```

Save Best Only

In [51]:
```python
del  model
keras.backend.clear_session()
```

In [52]:
```python
model = keras.models.Sequential([
    keras.layers.Dense(30, activation = "relu", input_shape=[8]),
    keras.layers.Dense(30, activation = "relu"),
    keras.layers.Dense(1)
])
```

In [53]:
```python
model.compile(loss = "mean_squared_error",
              optimizer = keras.optimizers.SGD(lr=1e-3),
              metrics = ["mae"])
```

In [54]:
```python
checkpoint_cb = keras.callbacks.ModelCheckpoint("Best_ Model.h5", save_best_only = True)
```

In [55]:
```python
history = model.fit(X_train, y_train, epochs = 10, validation_data =(X_valid, y_valid), callbacks = [checkpoint_cb])
```

```
Epoch 1/10
  1/363 [..............................] - ETA: 0s - loss: 5.4289 - mae: 2.0744WARNING:tensorflow:Callbacks method `on_train_batch_end` is slow compared to the batch time (batch time: 0.0000s vs `on_train_batch_end` time: 0.0011s). Check your callbacks.
363/363 [==============================] - 0s 901us/step - loss: 1.4581 - mae: 0.8737 - val_loss: 0.7883 - val_mae: 0.6579
Epoch 2/10
363/363 [==============================] - 0s 669us/step - loss: 0.7368 - mae: 0.6366 - val_loss: 0.7306 - val_mae: 0.5979
Epoch 3/10
363/363 [==============================] - 0s 705us/step - loss: 0.6516 - mae: 0.5904 - val_loss: 0.6501 - val_mae: 0.5580
Epoch 4/10
363/363 [==============================] - 0s 719us/step - loss: 0.5951 - mae: 0.5606 - val_loss: 0.5551 - val_mae: 0.5348
Epoch 5/10
363/363 [==============================] - 0s 918us/step - loss: 0.5536 - mae: 0.5376 - val_loss: 0.5025 - val_mae: 0.5091
Epoch 6/10
363/363 [==============================] - 0s 792us/step - loss: 0.5199 - mae: 0.5206 - val_loss: 0.4728 - val_mae: 0.4930
Epoch 7/10
363/363 [==============================] - 0s 694us/step - loss: 0.4940 - mae: 0.5059 - val_loss: 0.4497 - val_mae: 0.4821
Epoch 8/10
363/363 [==============================] - 0s 706us/step - loss: 0.4730 - mae: 0.4944 - val_loss: 0.4359 - val_mae: 0.4723
Epoch 9/10
363/363 [==============================] - 0s 781us/step - loss: 0.4563 - mae: 0.4856 - val_loss: 0.4267 - val_mae: 0.4658
Epoch 10/10
363/363 [==============================] - 0s 680us/step - loss: 0.4431 - mae: 0.4790 - val_loss: 0.4224 - val_mae: 0.4593
```

In [57]:
```python
model = keras.models.load_model("Best_ Model.h5")
mse_test = model.evaluate(X_test, y_test)
```

```
162/162 [==============================] - 0s 469us/step - loss: 0.4366 - mean_absolute_error: 0.4682
```

In [ ]: