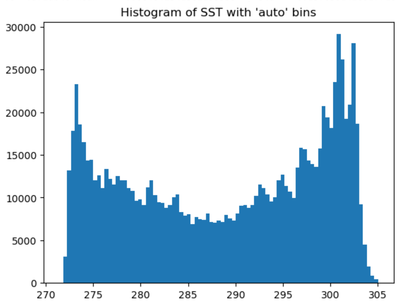
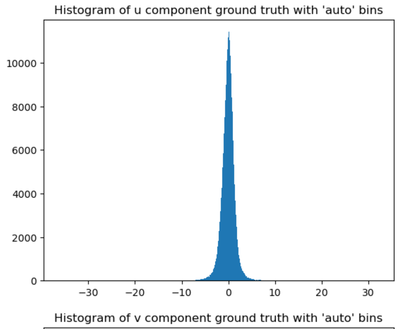
Connected to erasml_cnn (Python 3.11.9)

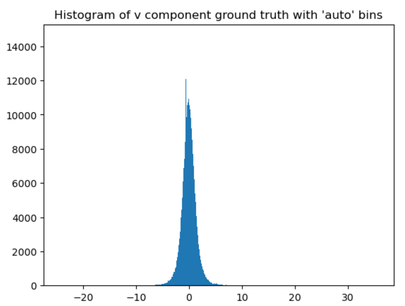
In [ ]:
```
from preprocessing import *


# We are going to perform a simple Exploratory Data Analysis on the ASCAT data. Very urgent to
# do so in order to align all of us with having clear the current problems to work on

import time
#import numpy as np
import matplotlib.pyplot as plt

start = time.time()

def generate_inputs_train(uNET_params, input_var_names, train_input_dir, eval_input_dir, np_train_files_dir, np_eval_files_dir, prefix):
    # Full model
    uNET_train_generator = uNETInputGenerator(input_var_names, target_var_names, input_dir=train_input_dir,
                                              output_dir=np_train_files_dir, downsample_ratio=1,
                                              records_per_file=1e10,
                                              seed=123, out_file_prefix=prefix)
    train_np_flist = uNET_train_generator.generate_np_files()
    uNET_eval_generator = uNETInputGenerator(input_var_names, target_var_names, input_dir=eval_input_dir,
                                              output_dir=np_eval_files_dir, downsample_ratio=1,
                                              records_per_file=1e10,
                                              seed=123, out_file_prefix=prefix)
    eval_np_flist = uNET_eval_generator.generate_np_files()
    train_data = np.load(train_np_flist[0])
    eval_data = np.load(eval_np_flist[0])

# Agreed on last meeting
input_var_names = ['lon', 'lat', 'eastward_model_wind', 'northward_model_wind', 'model_speed', 'model_dir',
                   'msl', 'air_temperature', 'q', 'sst', 'uo', 'vo']

print("Number of input variables / features = ",len(input_var_names))

scat_model_var_names = {'scat': ['eastward_wind', 'northward_wind'],
                        'model':['eastward_model_wind', 'northward_model_wind']}
target_var_names = ['u_diff', 'v_diff']
print("Number of output targets = ",len(target_var_names))

#File path env setup. Currently local execution for Adrian's iMac
# Train period from 02/01/2020 - 06/03/2020 both included
train_input_dir = "/Volumes/SSD Adrian/TFM/adrian_tfm/ASCAT_l3_collocations/2020/train/"
# Test period from  10/03/2020 - 01/05/2020 both included
eval_input_dir = "/Volumes/SSD Adrian/TFM/adrian_tfm/ASCAT_l3_collocations/2020/test/"

np_train_files_dir = "/Volumes/SSD Adrian/TFM/adrian_tfm/ASCAT_l3_collocations/2020/uNET_np_data/train/"
np_eval_files_dir = "/Volumes/SSD Adrian/TFM/adrian_tfm/ASCAT_l3_collocations/2020/uNET_np_data/test/"

plots_folder = "/Volumes/SSD Adrian/TFM/adrian_tfm/ASCAT_l3_collocations/2020/plots/uNET_importance/"
save_model_folder = "/Volumes/SSD Adrian/TFM/adrian_tfm/ASCAT_l3_collocations/2020/saved_models/"
file_prefix = "allvars_cpu_"

# Let's define some hyper-parameters
hparams = {
    'batch_size': 64,
    'num_epochs': 10,
    'test_batch_size': 64,
    'learning_rate': 1e-3,
    'log_interval': 100,
}

# Condition to generate the data only one time. A improvement may be to try to read folder and only generate
# when folders do not exist
if False:
    generate_inputs_train(hparams, input_var_names, train_input_dir, eval_input_dir, np_train_files_dir,
                          np_eval_files_dir, prefix=file_prefix)

train_fn = np_train_files_dir + "allvars_cpu_000.npz"
val_fn = np_eval_files_dir + "allvars_cpu_000.npz"

train_data = np.load(train_fn)
eval_data = np.load(val_fn)

# Now I can play freely with the data. It is from here when I am free to write the any code I want

# This is the unmasked train data (Input). In Shape = (995807, 12)
train = train_data['inputs']
# These are the input var names in a numpy.ndarray. Equivalent to the list input_var_names. Shape = (12,)
ix = train_data['input_var_names']
# This is the unmasked ground truth data (Input). Out Shape = (995807, 2)
ground_truth = train_data['targets']

for index in range(12):
    mean = np.nanmean(train[:,index]) #Computes ignoring NaN values
    std = np.nanstd(train[:,index]) #Computes ignoring NaN values
    maxd = train[:,index].max()
    mind = train[:,index].min()
    print(ix[index]," variable mean =  ",mean, " ; std = ",std," ; max = ",maxd," min = ",mind)


plt.hist(train[:,9], bins='auto')  # arguments are passed to np.histogram
plt.title("Histogram of SST with 'auto' bins")
plt.show()

aux = ['u','v']
for index in range(2):
    mean = np.nanmean(ground_truth[:,index]) #Computes ignoring NaN values
    std = np.nanstd(ground_truth[:,index]) #Computes ignoring NaN values
    maxd = ground_truth[:,index].max()
    mind = ground_truth[:,index].min()
    print(aux[index]," ground truth mean =  ",mean, " ; std = ",std," ; max = ",maxd," min = ",mind)

plt.hist(ground_truth[:,0], bins='auto')  # arguments are passed to np.histogram
plt.title("Histogram of u component ground truth with 'auto' bins")
plt.show()

plt.hist(ground_truth[:,1], bins='auto')  # arguments are passed to np.histogram
plt.title("Histogram of v component ground truth with 'auto' bins")
plt.show()

'''
plt.imshow(train_data['inputs'])
plt.colorbar()
plt.show()
'''
```

```
Number of input variables / features =  12
Number of output targets =  2
lon  variable mean =   187.28150591680918  ; std =  101.97836372099839  ; max =  359.9375  min =  0.0625
lat  variable mean =   -9.690244884299869  ; std =  40.60446266753369  ; max =  81.1875  min =  -78.3125
eastward_model_wind  variable mean =   0.5191599978710734  ; std =  6.423143242763846  ; max =  20.13  min =  -18.76
northward_model_wind  variable mean =   -0.27262090947342216  ; std =  5.650730693324713  ; max =  18.69  min =  -23.71
model_speed  variable mean =   7.82254986156956  ; std =  3.5183670559123534  ; max =  24.44  min =  0.08
model_dir  variable mean =   179.42139721853735  ; std =  99.04072014142872  ; max =  359.90000000000003  min =  0.1
msl  variable mean =   100958.16218403768  ; std =  1307.7704955222734  ; max =  103994.0  min =  96317.0
air_temperature  variable mean =   287.51316891727  ; std =  10.508818000336138  ; max =  304.1  min =  241.1
q  variable mean =   0.009365524534372621  ; std =  0.005572659424648705  ; max =  0.02196  min =  0.00018
sst  variable mean =   289.03304673887953  ; std =  10.186094154068178  ; max =  nan  min =  nan
uo  variable mean =   0.015420961247933768  ; std =  0.19631212928273892  ; max =  nan  min =  nan
vo  variable mean =   0.004841341362189539  ; std =  0.1576391426927716  ; max =  nan  min =  nan
```




```
u  ground truth mean =   -0.05095778599684116  ; std =  1.4594783526702045  ; max =  32.07  min =  -36.16
v  ground truth mean =   -0.017351314059852962  ; std =  1.5047666260707266  ; max =  35.7  min =  -24.45
```





Out[ ]:
```
"\nplt.imshow(train_data['inputs'])\nplt.colorbar()\nplt.show()\n"
```