A Appendix

Genetics of Significant Celiac Disease SNPs

SNP	Location	Allele	Gene	Most severe consequence	Affected Traits
rs3891175	Chr6:32666690	C/T	HLA-DQB1	5 prime utr variant	celiac disease, aspartate aminotransferase measurement, autoimmune hepatitis
rs1265754	Chr6:32335915	T/A	TSBP1	missense variant	inguinal hernia, celiac disease
rs9273529	Chr6:32628698	C/T	HLA-DQB1	Intron variant	Rheumatoid arthritis, Hypothyroidism, celiac disease
rs9274253	Chr6:32631348	G/A	HLA-DQB1	Intron variant	Rheumatoid arthritis, Hypothyroidism, Thyrotoxicosis, celiac disease, diabetes
rs9267488	Chr6:31546470	G/A	ATP6V1G2	Splice region variant	Myositis, inguinal hernia, intelligence, Malabsorption/coeliac disease, gastrointestinal disease
rs204989	Chr6:32161852	G/A	GPSM3	Intron variant	rheumatoid arthritis, ulcerative colitis
gender	N/A	N/A	N/A	N/A	N/A
rs1383264	Chr6:32739967	A/T	HLA-DQB2	Intron variant	diabetes, psoriasis, celiac
rs9469220	Chr6:32658310	G/A	Non-coding between HLA-DQB1 and HLA-DQA2	N/A	Chron's disease
rs10133464	14-97897269	T/C	N/A	N/A	unknown function in genome

Table 8: Significant Celiac Disease SNPs and their associated location, allele (major/ minor), gene, most severe mutation caused by variant, and affected traits as determined in the literature. Variants with N/A values had attributes that were not found in the literature. Additionally, gender has N/A values for their genomic descriptions. We did not include PCs as variants since we did not want to capture demographic differences in phenotypes.

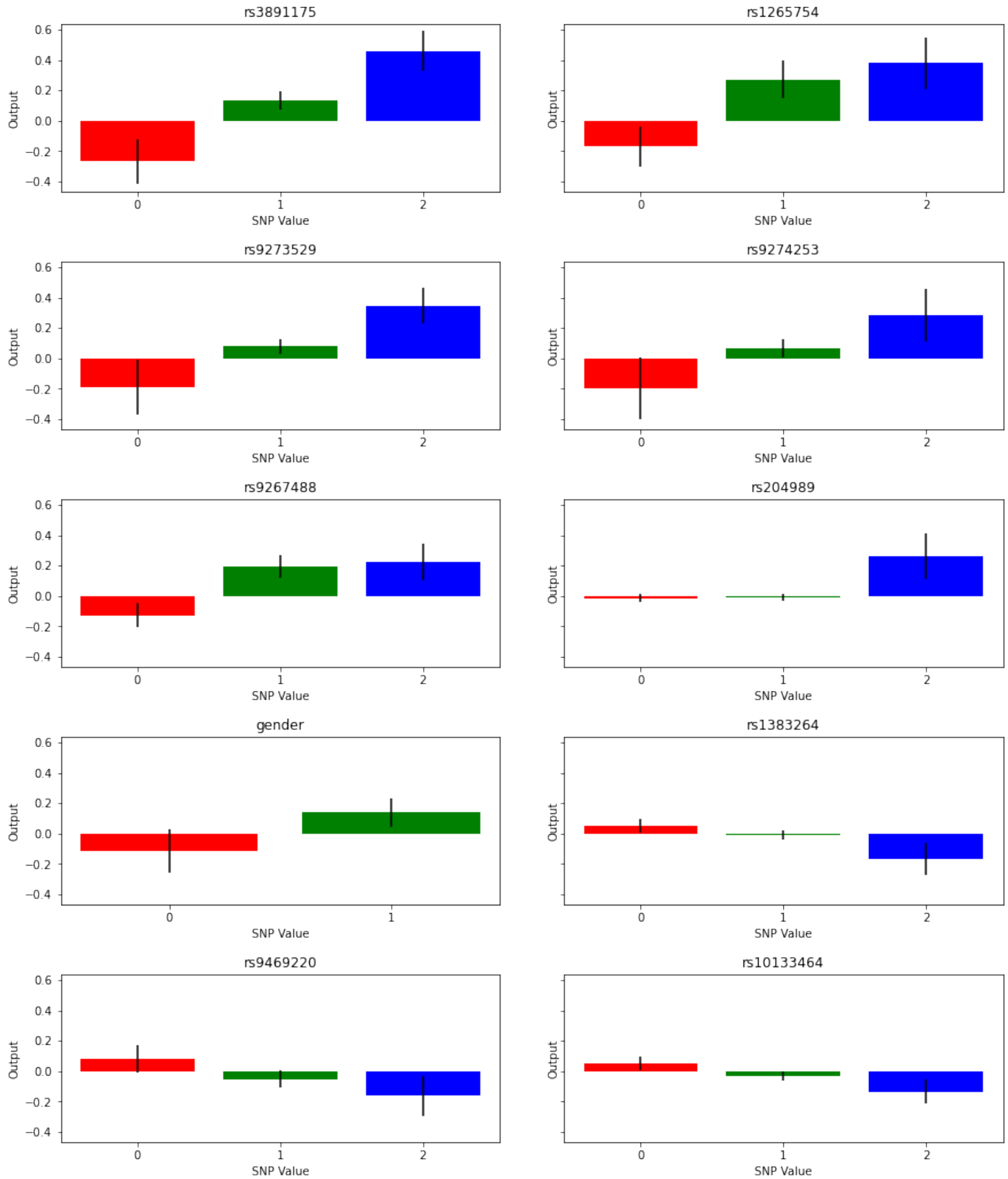


Figure 3: Allelic Contributions for the top 10 most important SNPs for celiac disease. Alleles (0 copies of alternate, 1 copy of alternate (heterozygote), 2 copies of alternate) are listed on the x axis, and contribution to model output is listed on the y axis.

Genetics of Significant Total Bilirubin SNPs

SNP	Location	Allele	Gene	Most severe consequence	Affected Traits
rs6742078	Chr2:233763993	G/T	UGT1A1, UGT1A3, UGT1A4, UGT1A5, UGT1A6, UGT1A7, UGT1A8, UGT1A9, UGT1A10	intron variant	serum metabolite measurement, bilirubin measurement, response to tenofovir, blood protein measurement, bilirubin measurement x insomnia, bilirubin measurement x response to tenofovir, aldosterone measurement, circulating cell free DNA measurement
rs887829	Chr2:233759924	C/T	UGT1A3, UGT1A4, UGT1A5, UGT1A6, UGT1A7, UGT1A8, UGT1A9, UGT1A10	intron variant	metabolite measurement, serum metabolite measurement, bilirubin measurement, blood metabolite measurement, biliverdin measurement, bilirubin measurement x insomnia, X-11530 measurement, bilirubin measurement x response to tenofovir, total cholesterol measurement, blood protein measurement, cholelithiasis x bilirubin measurement
rs34622615	Chr2:233743662	G/T	DNAJB3, UGT1A3, UGT1A4, UGT1A5, UGT1A6, UGT1A7, UGT1A8, UGT1A9, UGT1A10	non coding transcript exon variant	bilirubin measurement, total cholesterol measurement
rs4148325	Chr2:233764663	C/T	UGT1A1, UGT1A3, UGT1A4, UGT1A5, UGT1A6, UGT1A7, UGT1A8, UGT1A9, UGT1A10	intron variant	metabolite measurement, bilirubin measurement, serum metabolite measurement, biliverdin measurement, bilirubin measurement x response to tenofovir, blood protein measurement, biliverdin measurement, xanthurenate measurement
rs1661052	Chr11:2921363	A/G	SLC22A18	noncoding transcript exon variant	bilirubin measurement, total cholesterol measurement, high density lipoprotein cholesterol measurement, calcium measurement, apolipoprotein a1measurement
Gender	N/A	N/A	N/A	N/A	N/A
rs2070959	Chr2:233693545	A/G	UGT1A6, UGT1A7, UGT1A8, UGT1A9, UGT1A10	missense variant	bilirubin measurement x insomnia, gallstones, bilirubin measurement
rs34691116	Chr12:20874393	C/T	SLC01B3-SLC01B7, SLC01B3	intron variant	serum gamma-glutamyl transferase measurement, bilirubin measurement
rs4124874	Chr2:234665659	G/T	UGT1A3, UGT1A4, UGT1A5, UGT1A6, UGT1A7, UGT1A8, UGT1A9, UGT1A10	intron variant	bilirubin measurement
rs11045819	Chr12:21176879	C/A	SLC01B1	missense variant	lysophosphatidylethanolamine measurement, urate measurement, bilirubin measurement

Table 9: Top 10 most significant Total Bilirubin SNPs and their associated location, allele (major/minor), gene, most severe mutation caused by variant, and affected traits as determined in the literature

Genetics of Significant Age Diabetes Diagnosed SNPs

SNP	Location	Allele	Gene	Most severe consequence	Affected Traits
rs9273363	Chr6:32658495	C/A	HLA-DQA1, HLA-DQB1	regulatory region variant	type 1 diabetes mellitus, chromic lymphocytic leukemia
rs3916765	Chr6:32717773	G/A	HLA-DQB3, MTC03P1	intergenic variant	acute myeloid leukemia, type 2 diabetes mellitus
rs3842752	Chr11:2159843	G/A	INS-IGF2, INS	missense variant	HbA1c measurement, sex hormone-binding globulin measurement, rate measurement, IgF-1 measurement, serum urea measurement, creatinine measurement, glucose measurement, glomerular filtration rate
rs6034239	Chr20:1616137	G/A	SIRPG, RP11-77C3.3	missense variant	blood protein measurement, anti-meningococcal C serum bactericidal antibody measurement response to vaccine, mean platelet volume, type 1 diabetes mellitus, schizophrenia, lymphocyte percentage of leukocytes, mitochondrial DNA measurement, platelet component distribution width, platelet grit, brain aneurysm, neurofibrillary tangles measurement
rs6356	Chr11:2169721	C/T	TH	missense variant	sex hormone-binding globulin measurement
rs805304	Chr6:31698088	T/G	DDAH2	5' UTR variant	BMI-adjusted hip circumference, serum levels of protein C6orf2, feeling nervous
rs3129871	Chr6:32438565	C/A	TSBP1-AS1, HLA-DRA	intergenic variant	multiple sclerosis, multiple sclerosis x ogliclonal band measurement
Affx-52353201	N/A	N/A	N/A	N/A	N/A
rs6822933	Chr4:54059445	A/G	SCFD2	intron variant	neuroimaging measurement, white matter integrity, cortical surface area measurement, uterine fibroid, brain measurement, lipid measurement, DNA methylation, insomnia, balding measurement, breast carcinoma, medial orbital frontal cortex volume measurement, irritable bowel syndrome symptom measurement, intraocular pressure measurement, open-angle glaucoma, cytokine measurement, mean fractional anisotropy measurement, FEV/FEC ratio, myeloid white cell count, alanine measurement, neutrophil count, leukocyte count, colorectal health, cortical surface area measurement x neuroimaging measurement, optic disk size measurement, smoking behavior measurement, platelet count, PHF-tau measurement, testosterone measurement, calcium measurement, platelet crit, disease progression measurement x metastasis measurement, cataract
rs1043618	Chr6:31815730	G/C	HSPA1A	5' UTR variant	cataract age at diagnosis, type 2 diabetes

Table 10: Significant Age Diabetes Diagnosed SNPs and their associated location, allele (major/minor), gene, most severe mutation caused by variant, and affected traits as determined in the literature

Genetics of Significant Red Hair SNPs

SNP	Location	Allele	Gene	Most severe consequence	Affected Traits
rs11547464	Chr16:89919683	G/A	MC1R	Missense variant	hair color, hair color measurement
Affx-35293625	Chr16:89986117	T/C	DEF8	Missense variant	skin of upper limb, skin of scalp and neck, malignant melanoma of upper limb, malignant melanoma of lower limb, Actinic keratosis
rs1805009	Chr16:89919683	G/A	MC1R	Missense variant	hair color, hair color measurement
affx-80298222	Chr16:89986202	T/A	N/A	N/A	N/A
rs1805006	Chr16:89919510	C/A	MC1R	Missense variant	hair color
rs58208647	Chr16:89730629	A/G	SPATA33	Intron variant	puberty onset (male), hair color, hair color measurement, hemoglobin measurement, BMI-adjusted hip circumference
1800347	Chr16:89748641	T/C	FANCA	Intron variant	hair color
rs56288641	Chr16:89777078	G/A	VPS9D1	missense variant	Skin color, hair color, ease of skin tanning
rs117204628	Chr16:89966047	C/T	DEF8	3' utr variant	hair color, keratinocyte carcinoma, basal cell carcinoma

Table 11: Significant Red Hair SNPs and their associated location, allele (major/ minor), gene, most severe mutation caused by variant, and affected traits as determined in the literature. Variants with N/A values had attributes that were not found in the literature. Additionally, gender has N/A values for their genomic descriptions. We did not include PCs as variants since we did not want to capture demographic differences in phenotypes.

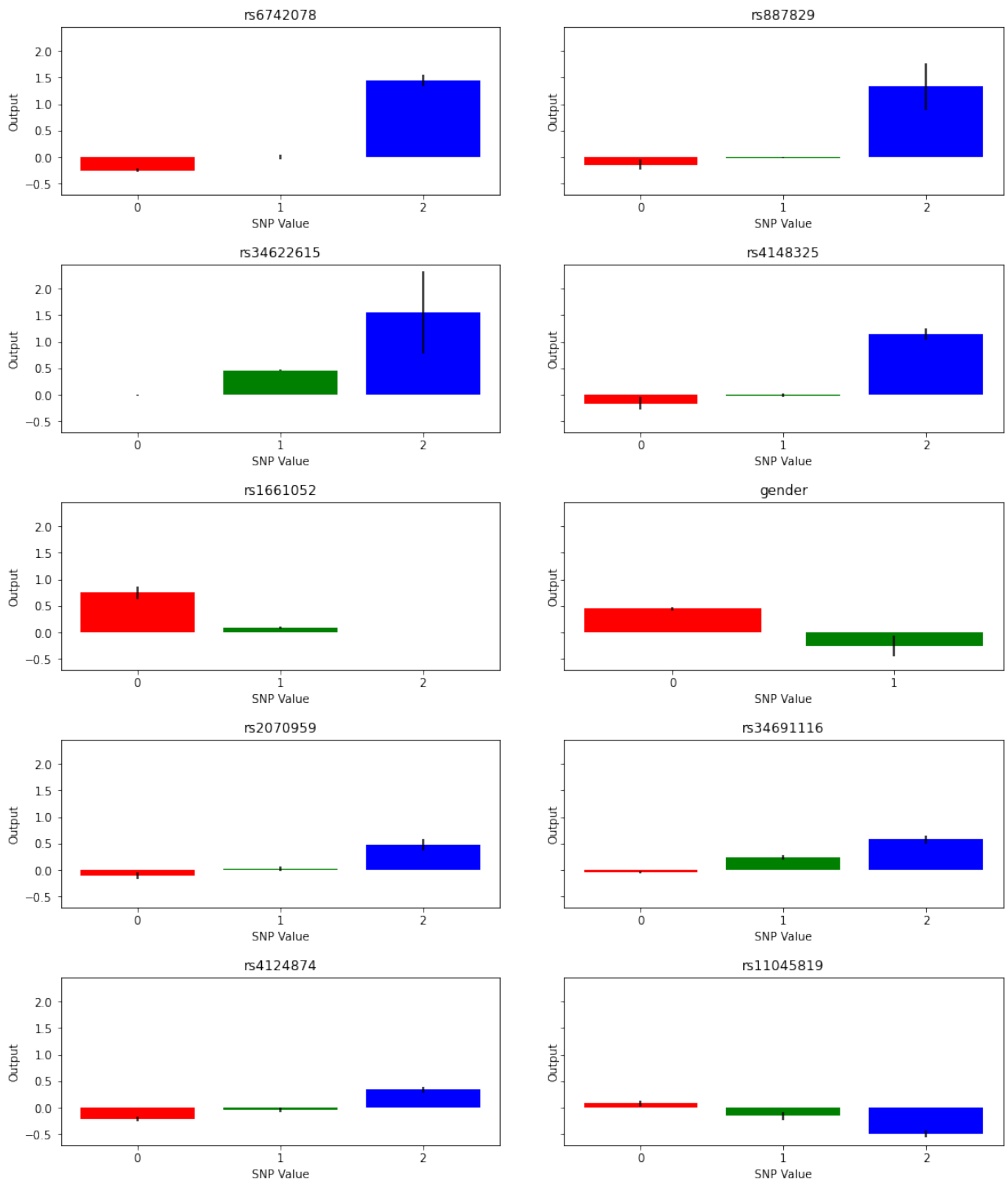


Figure 4: Allelic Contributions for the top 10 most important SNPs for total bilirubin levels. Alleles (0 copies of alternate, 1 copy of alternate (heterozygote), 2 copies of alternate) are listed on the x axis, and contribution to model output is listed on the y axis.

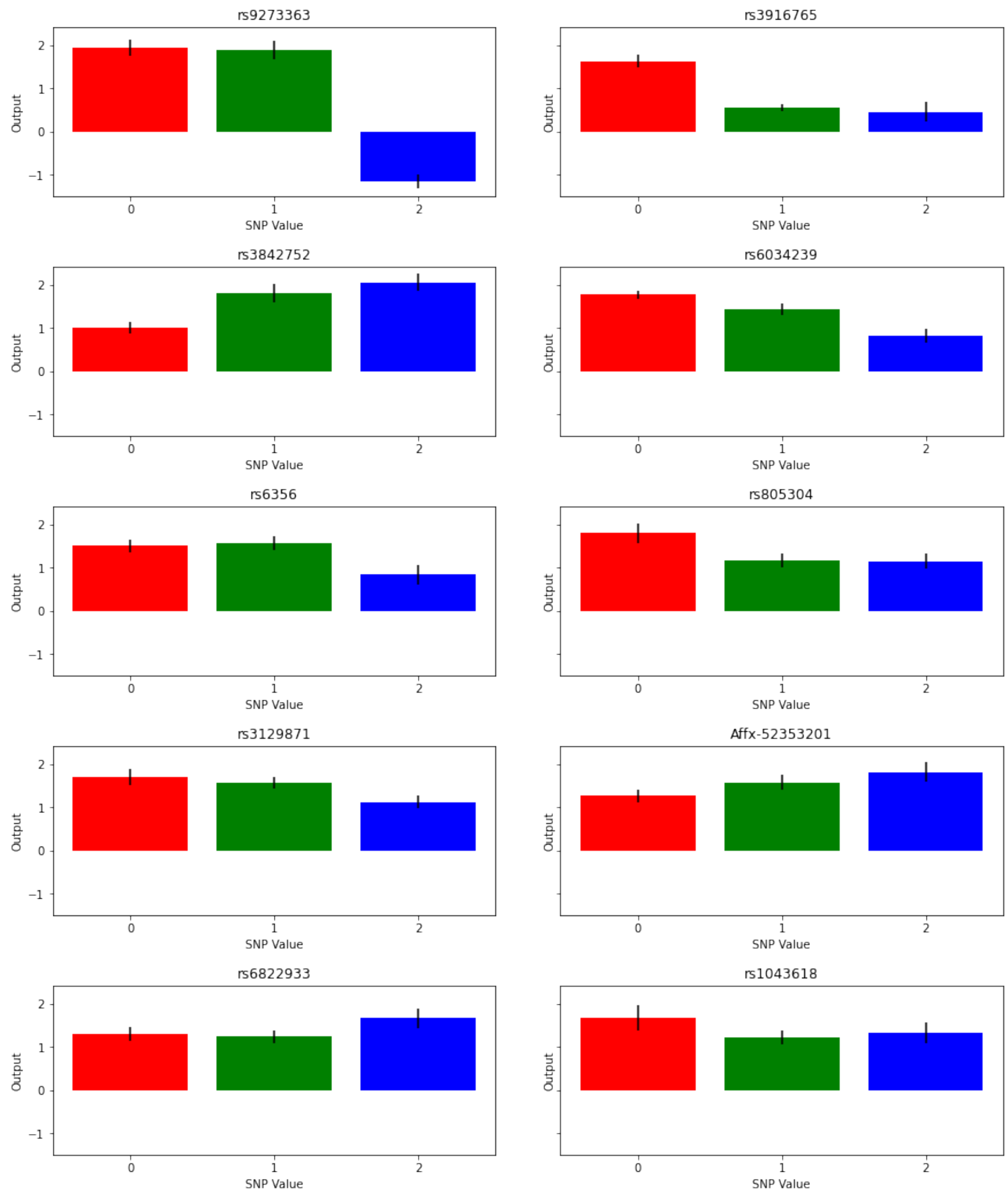


Figure 5: Allelic Contributions for the top 10 most important SNPs for age at diabetes diagnosis. Alleles (0 copies of alternate, 1 copy of alternate (heterozygote), 2 copies of alternate) are listed on the x axis, and contribution to model output is listed on the y axis.

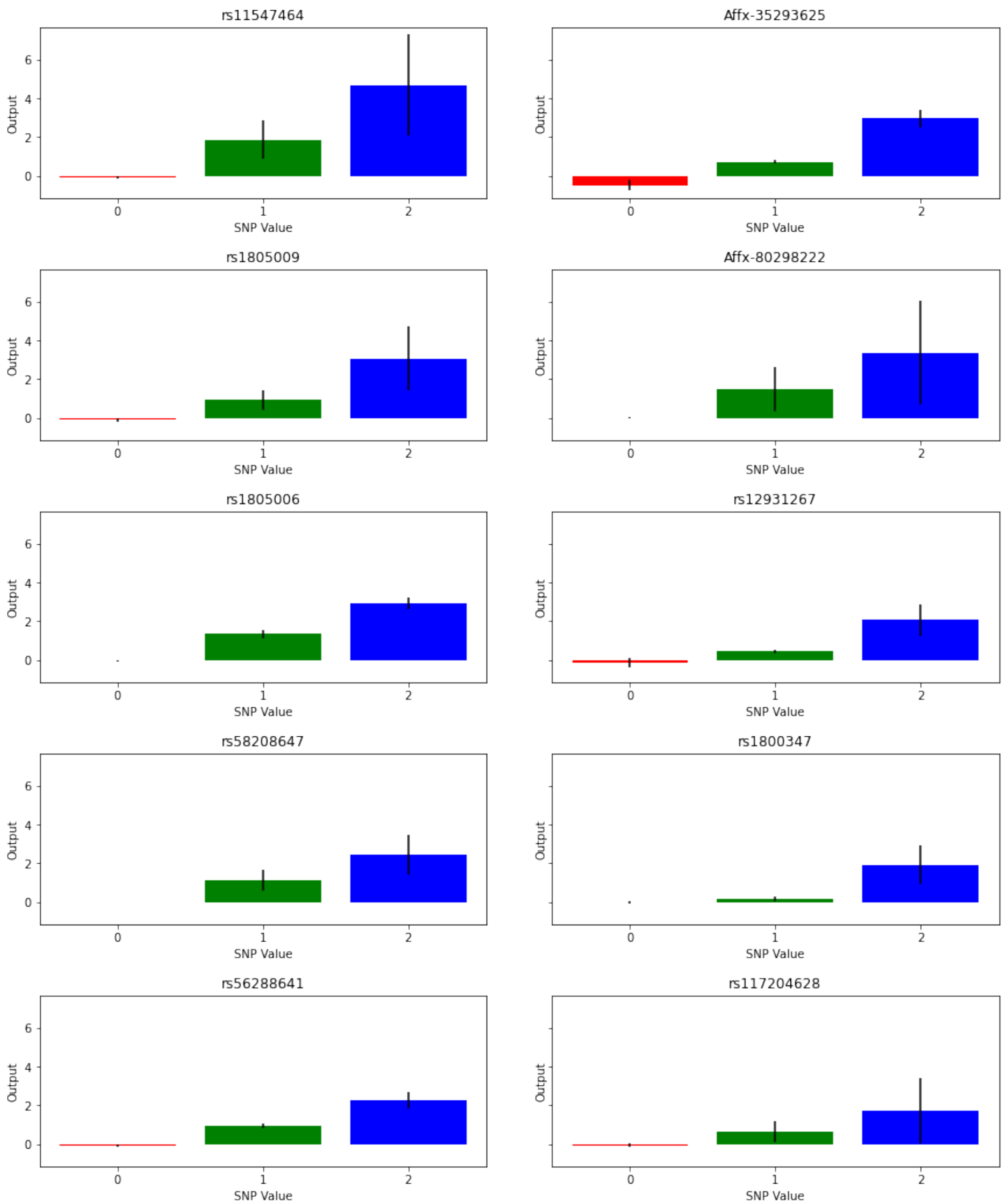


Figure 6: Allelic Contributions for the top 10 most important SNPs for red hair. Alleles (0 copies of alternate, 1 copy of alternate (heterozygote), 2 copies of alternate) are listed on the x axis, and contribution to model output is listed on the y axis.

In [1]:
```
import numpy as np
import os
import pandas as pd
import random
import pgenlib as pg
```

First, get the individual IDs and their phenotypes for given phenotypes of interest

In [2]:
```
# info_df contains all the phenotypes in the ukb
info_f = 'phenotypes/phenotype_info.tsv'
info_df = pd.read_csv(info_f, sep='\t')
info_df.head(5)
```

Out[2]:

	#GBE_ID	GBE_NAME	FIELD	TABLE	BASKET	APP_ID	N	N_GBE	N_NBW	N_AFR	N_EAS	N_SAS	N_SMR
0	BIN100020	Typical_diet_yesterday	100020	37855.0	2005693.0	24983	192965	133748	10426	1857	409	2137	16346
1	BIN100240	Coffee_consumed	100240	37855.0	2005693.0	24983	161624	113691	8832	962	276	1117	13674
2	BIN100260	Added_milk_to_instant_coffee_always_(Diet,_24h...	100260	37855.0	2005693.0	24983	97522	69808	3992	606	169	737	8598
3	BIN100280	Added_milk_to_filtered_coffee_always_(Diet,_24...	100280	37855.0	2005693.0	24983	42917	30404	2747	96	61	173	3446
4	BIN10030500	Microalbumin_higher_than_40_mg/L	NaN	NaN	NaN	24983	36858	23590	1842	933	89	990	3301

In [3]:
```
# master_df contains phenotype information for each individual in the ukb
master_f = 'phenotypes/master.20211020.phe'
master_df = pd.read_csv(master_f, sep='\t')
master_df
```

Out[3]:

	#FID	IID	population	split	split_nonWB	age	age0	age1	age2	age3	...	INI25722	INI25723	BIN_FC
0	-1.0	-1.0	DO_NOT_PASS_SQC	DO_NOT_PASS_SQC	NaN	67	NaN	NaN	NaN	NaN	...	-9.000000	-9.000000	
1	-2.0	-2.0	DO_NOT_PASS_SQC	DO_NOT_PASS_SQC	NaN	67	NaN	NaN	NaN	NaN	...	-9.000000	-9.000000	
2	-3.0	-3.0	DO_NOT_PASS_SQC	DO_NOT_PASS_SQC	NaN	67	NaN	NaN	NaN	NaN	...	-9.000000	-9.000000	
3	-4.0	-4.0	DO_NOT_PASS_SQC	DO_NOT_PASS_SQC	NaN	67	NaN	NaN	NaN	NaN	...	-9.000000	-9.000000	
4	-5.0	-5.0	DO_NOT_PASS_SQC	DO_NOT_PASS_SQC	NaN	67	NaN	NaN	NaN	NaN	...	-9.000000	-9.000000	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
516764	6026202.0	6026202.0	white_british	test	NaN	54	46.394521	NaN	NaN	NaN	...	-9.000000	-9.000000	
516765	6026216.0	6026216.0	white_british	train	NaN	56	47.413699	NaN	57.158904	NaN	...	0.160961	0.057493	
516766	6026229.0	6026229.0	related	related	train	77	67.652055	NaN	NaN	NaN	...	-9.000000	-9.000000	
516767	6026237.0	6026237.0	white_british	test	NaN	77	70.161644	NaN	NaN	NaN	...	-9.000000	-9.000000	
516768	6026241.0	6026241.0	-9	-9	NaN	-9	-9.000000	-9.0	-9.000000	-9.0	...	-9.000000	-9.000000	

516769 rows × 3511 columns

In [4]:
```
# Chosen Phenotypes
# BIN_FC2001747: red hair color; INI30840: total bilirubin
chosen_phe = ['INI30790']
phe_mapper = {x: i for i, x in enumerate(chosen_phe)}
phe_mapper
```

Out[4]: {'INI30790': 0}

In [5]:
```
# Obtain the relevant info_df rows with the given phenotypes
chosen_phe_info_df = info_df[info_df['#GBE_ID'].isin(chosen_phe)].sort_values(by='#GBE_ID', key=lambda x: x.map(phe_ma
chosen_phe_info_df
```

Out[5]:

	#GBE_ID	GBE_NAME	FIELD	TABLE	BASKET	APP_ID	N	N_GBE	N_NBW	N_AFR	N_EAS	N_SAS	N_SMR	N_OTH	SOURCE	DATE	
0	INI30790	Lipoprotein_A	30790	37855.0	2005693.0	24983	377672	257047	19197	5086	994	6590	34071	22383	ukb_annotations.tsv	2020-06-18	/(

In [6]:
```
# individuals with valid chosen phenotype data
# CHANGED from Yoko and Haya's original code - only removed patients with BOTH bilirubin and hair phenotypes missing
chosen_phe_df = master_df[['#FID', 'IID']+chosen_phe].replace(-9, np.nan)
chosen_phe_df['#FID'] = chosen_phe_df['#FID'].apply(lambda x: str(int(x)))
chosen_phe_df['IID'] = chosen_phe_df.IID.apply(lambda x: int(x))
chosen_phe_df = chosen_phe_df.dropna(subset=chosen_phe, how='all')
chosen_phe_df
```

Out[6]:

	#FID	IID	INI30790
9	1000034	1000034	4.11
10	1000045	1000045	16.48
11	1000052	1000052	49.80
12	1000069	1000069	79.90
13	1000076	1000076	80.10
...	...	...	...
516763	6026191	6026191	3.99
516764	6026202	6026202	11.30
516765	6026216	6026216	4.42
516766	6026229	6026229	11.75
516767	6026237	6026237	183.70

377661 rows × 3 columns

In [7]:
```
# Some of the resulting individuals have either bilirubin or hair phenotype missing, but not both
chosen_phe_df.isnull().sum()
```

Out[7]:
```
#FID        0
IID         0
INI30790    0
dtype: int64
```

In [10]:
```
# Find the individuals with SNP data also available
# Question: what is this fam file and why is it relevant??????
fam_file = 'ukb/ukb24983_cal_cALL_v2_hg19.fam'
fam_data = pd.read_csv(fam_file, sep='\t', names=['fid', 'iid', 'father', 'mother', 'gender', 'trait'])
IDs = set(chosen_phe_df.IID).intersection(set(fam_data.iid))
snp_chosen_phe_df = chosen_phe_df[chosen_phe_df['IID'].isin(IDs)]
print(snp_chosen_phe_df.shape)
```

```
(374221, 3)
```

In [12]:
```
snp_chosen_phe_df = snp_chosen_phe_df.sort_values(by=['IID']).reset_index(drop=True)
#snp_chosen_phe_df['BIN_FC2001747'] = snp_chosen_phe_df['BIN_FC2001747'].replace([1.0, 2.0], [0, 1])
snp_chosen_phe_df
```

Out[12]:

	#FID	IID	INI30790
0	1000034	1000034	4.11
1	1000045	1000045	16.48
2	1000052	1000052	49.80
3	1000069	1000069	79.90
4	1000076	1000076	80.10
...	...	...	...
374216	6026191	6026191	3.99
374217	6026202	6026202	11.30
374218	6026216	6026216	4.42
374219	6026229	6026229	11.75
374220	6026237	6026237	183.70

374221 rows × 3 columns

In [13]:
```
# Function to find the class balance and positive/negative IDs for a given binary phenotype
# Returns a list of the positive IDs and a list of the negative IDs
def class_balance(phenotype):
    negativeIDs = list(snp_chosen_phe_df[snp_chosen_phe_df['BIN_FC2001747'] == 0].IID)
    positiveIDs = list(snp_chosen_phe_df[snp_chosen_phe_df['BIN_FC2001747'] == 1].IID)
    print("Phenotype: {}".format(phenotype))
    print("Number of positive samples: {}".format(len(positiveIDs)))
    print("Number of negative samples: {}".format(len(negativeIDs)))
    return positiveIDs, negativeIDs
```

In [14]:
```
# Class balance for the binary red hair phenotype: BIN_FC2001747
#red_hair_posIDs, red_hair_neg_IDs = class_balance('BIN_FC2001747')
```

In [20]:
```
# Saves the relevant IDs and phenotype values in a text file
def save_phenotypes(phenotype):
    df_copy = snp_chosen_phe_df.copy().loc[:, ['IID', phenotype]]
    subset = df_copy.dropna(subset=[phenotype])
    subset.to_csv('phenotypes/{}_phenotypes.tsv'.format(phenotype), sep='\t', index=False, header=True)
```

In [21]:
```
# Save the IDs  and phenotypes for the red hair phenotype
#save_phenotypes('BIN_FC2001747')

# Save the IDs  and phenotypes for the bilirubin phenotype
#save_phenotypes('INI30840')

save_phenotypes('INI30790')
```

How to UKB

In [22]:
```
file_root = 'ukb/ukb24983_cal_cALL_v2_hg19'
bim_file = f'{file_root}.bim'
bim_data = pd.read_csv(bim_file, sep='\t', names=['chrom', 'snp', 'cm', 'pos', 'a0', 'a1'], header=None, low_memory=Fa
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
/tmp/ipykernel_663389/4108118851.py in <module>
      1 file_root = 'ukb/ukb24983_cal_cALL_v2_hg19'
      2 bim_file = f'{file_root}.bim'
----> 3 bim_data = pd.read_csv(bim_file, sep='\t', names=['chrom', 'snp', 'cm', 'pos', 'a0', 'a1'], header=None, low
_memory=False).reset_index()

~/anaconda3/lib/python3.7/site-packages/pandas/util/_decorators.py in wrapper(*args, **kwargs)
    309                     stacklevel=stacklevel,
    310                 )
--> 311             return func(*args, **kwargs)
    312 
    313         return wrapper

~/anaconda3/lib/python3.7/site-packages/pandas/io/parsers/readers.py in read_csv(filepath_or_buffer, sep, delimiter,
header, names, index_col, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, true_values, false_
values, skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbose, skip_blank_li
nes, parse_dates, infer_datetime_format, keep_date_col, date_parser, dayfirst, cache_dates, iterator, chunksize, com
pression, thousands, decimal, lineterminator, quotechar, quoting, doublequote, escapechar, comment, encoding, encodi
ng_errors, dialect, error_bad_lines, warn_bad_lines, on_bad_lines, delim_whitespace, low_memory, memory_map, float_p
recision, storage_options)
    584     kwds.update(kwds_defaults)
    585 
--> 586     return _read(filepath_or_buffer, kwds)
    587 
    588 

~/anaconda3/lib/python3.7/site-packages/pandas/io/parsers/readers.py in _read(filepath_or_buffer, kwds)
    480 
    481     # Create the parser.
--> 482     parser = TextFileReader(filepath_or_buffer, **kwds)
    483 
    484     if chunksize or iterator:

~/anaconda3/lib/python3.7/site-packages/pandas/io/parsers/readers.py in __init__(self, f, engine, **kwds)
    809             self.options["has_index_names"] = kwds["has_index_names"]
    810 
--> 811         self._engine = self._make_engine(self.engine)
    812 
    813     def close(self):

~/anaconda3/lib/python3.7/site-packages/pandas/io/parsers/readers.py in _make_engine(self, engine)
   1038             )
   1039         # error: Too many arguments for "ParserBase"
-> 1040         return mapping[engine](self.f, **self.options)  # type: ignore[call-arg]
   1041 
   1042     def _failover_to_python(self):

~/anaconda3/lib/python3.7/site-packages/pandas/io/parsers/c_parser_wrapper.py in __init__(self, src, **kwds)
     49 
     50         # open handles
---> 51         self._open_handles(src, kwds)
     52         assert self.handles is not None
     53 

~/anaconda3/lib/python3.7/site-packages/pandas/io/parsers/base_parser.py in _open_handles(self, src, kwds)
    227             memory_map=kwds.get("memory_map", False),
    228             storage_options=kwds.get("storage_options", None),
--> 229             errors=kwds.get("encoding_errors", "strict"),
    230         )
    231 

~/anaconda3/lib/python3.7/site-packages/pandas/io/common.py in get_handle(path_or_buf, mode, encoding, compression,
memory_map, is_text, errors, storage_options)
    705                 encoding=ioargs.encoding,
    706                 errors=errors,
--> 707                 newline="",
    708             )
    709         else:

FileNotFoundError: [Errno 2] No such file or directory: 'ukb/ukb24983_cal_cALL_v2_hg19.bim'
```

In [15]:
```
bim_data
```

Out[15]:

	index	chrom	snp	cm	pos	a0	a1
0	0	1	rs28659788	0	723307	G	C
1	1	1	rs116587930	0	727841	A	G
2	2	1	rs116720794	0	729632	T	C
3	3	1	rs3131972	0	752721	G	A
4	4	1	rs12184325	0	754105	T	C
...	...	...	...	...	...	...	...
805421	805421	MT	Affx-92047842	0	16337	T	C
805422	805422	MT	Affx-79443531	0	16356	C	T
805423	805423	MT	Affx-79443532	0	16362	C	T
805424	805424	MT	Affx-89025709	0	16390	A	G
805425	805425	MT	Affx-79381726	0	16391	A	G

805426 rows × 7 columns

In [16]:
```
# Get the relevant SNPs for the given phenotype
def getSnpIdxs(phenotype, bim_data):
    # Merge the PRS weights for each phenotype with the bim file to find the corresponding SNP indices for this phenot
    prs_weights= pd.read_csv("phenotypes/{}.snpnetBETAs.tsv".format(phenotype), sep='\t').sort_values(by='BETA', ascen
    prs = pd.merge(bim_data, prs_weights, left_on='snp', right_on='ID')['index'].tolist()
    print('There are {} SNPs for phenotype {}'.format(len(prs), phenotype))
    snpIdxs = np.array(sorted(prs)).astype(np.uint32)
    return snpIdxs
```

In [17]:
```
# QUESTION: in Yoko and Haya's code, they have 100000 total samples for hair (downsampled), why??????
hair_snp_idxs = getSnpIdxs('BIN_FC2001747', bim_data)
bilirubin_snp_idxs = getSnpIdxs('INI30840', bim_data)
```

```
There are 1621 SNPs for phenotype BIN_FC2001747
There are 1159 SNPs for phenotype INI30840
```

In [18]:
```
# Gets the patient IDs for each phenotype from the corresponding file saved earlier in the notebook
# If the phenotype is binary, then downsample the number of negative samples so that the classes are balanced
def getSampleIDs(phenotype, binary):
    file = "phenotypes/{}_phenotypes.tsv".format(phenotype)
    df = pd.read_csv(file, sep='\t')
    # If it is a binary phenotype, then downsample negative samples if necessary
    sampleIDs = []
    if binary:
        posIDs = list(df[df[phenotype] == 1]['IID'])
        negIDs = list(df[df[phenotype] == 0]['IID'])
        # make the # negative samples = # positive samples
        negIDs_sampled = list(random.sample(negIDs, len(posIDs)))
        sampleIDs = posIDs + negIDs_sampled
    else:
        sampleIDs = list(df['IID'])
    return sampleIDs
```

In [19]:
```
# Get the SNPs for each relevant participant and each relevant SNP for a given phenotype
def saveSNPs(sampleIDs, phenotype, binary, variant_idxs, plink_file, out_path):
    sample_idxs = np.array(sorted(fam_data[fam_data['iid'].isin(sampleIDs)].index.tolist())).astype(np.uint32)
    # Read in the plink file
    data = pg.PgenReader(plink_file, raw_sample_ct=fam_data.shape[0], sample_subset=sample_idxs) #488377
    geno_mat_ukb = np.ascontiguousarray(np.zeros((len(sample_idxs), len(variant_idxs))).astype(np.int8).T)
    print('Reading data...')
    data.read_list(variant_idxs, geno_mat_ukb)
    print('Transposing...')
    # geno_mat_ukb contains all SNPs
    geno_mat_ukb = geno_mat_ukb.T
    print(geno_mat_ukb.shape, geno_mat_ukb.mean())
    np.save(out_path, geno_mat_ukb)
```

In [20]:
```
plink_file = b'/scratch/users/jlhought/ukb/ukb24983_cal_cALL_v2_hg19.bed'
hair_sample_IDs = sorted(getSampleIDs('BIN_FC2001747', True))
saveSNPs(hair_sample_IDs, 'BIN_FC2001747', True, hair_snp_idxs, plink_file, 'ukb/ukb24983_cal_cALL_v2_hg19_SUBSET_PRS_H
bili_sample_IDs = sorted(getSampleIDs('INI30840', False))
saveSNPs(bili_sample_IDs, 'INI30840', False, bilirubin_snp_idxs, plink_file, 'ukb/ukb24983_cal_cALL_v2_hg19_SUBSET_PRS_
```

```
Reading data...
Transposing...
(42016, 1621) 0.8342694454872337
Reading data...
Transposing...
(464659, 1159) 0.831463551622011
```

In [23]:
```
geno_data_hair = pd.DataFrame(np.load('ukb/ukb24983_cal_cALL_v2_hg19_SUBSET_PRS_HAIR.npy'))
phenotypes_hair_all = pd.read_csv('phenotypes/BIN_FC2001747_phenotypes.tsv', sep='\t')
# Take the downsampled version of phebotypes_hair_all
phenotypes_hair_subset = phenotypes_hair_all[phenotypes_hair_all['IID'].isin(hair_sample_IDs)]

geno_data_bili = pd.DataFrame(np.load('ukb/ukb24983_cal_cALL_v2_hg19_SUBSET_PRS_BILI.npy'))
phenotypes_bili = pd.read_csv('phenotypes/INI30840_phenotypes.tsv', sep='\t')
```

In [24]:
```
# Get the sample IDs so that gender can be added to the X values
hair_gender = fam_data[fam_data['iid'].isin(hair_sample_IDs)].reset_index()[['iid', 'gender']]
hair_gender.loc[:,'gender'] = hair_gender['gender'].replace(to_replace=1, value=0)
hair_gender.loc[:,'gender'] = hair_gender['gender'].replace(to_replace=2, value=1)

bili_gender = fam_data[fam_data['iid'].isin(bili_sample_IDs)].reset_index()[['iid', 'gender']]
bili_gender.loc[:,'gender'] = bili_gender['gender'].replace(to_replace=1, value=0)
bili_gender.loc[:,'gender'] = bili_gender['gender'].replace(to_replace=2, value=1)
```

In [46]:
```
hair_snps = pd.concat([hair_gender, geno_data_hair], axis=1).set_index('iid')
bili_snps = pd.concat([bili_gender, geno_data_bili], axis=1).set_index('iid')
```

In [54]:
```
hair_data = hair_snps.merge(phenotypes_hair_subset, left_on='iid', right_on='IID').set_index('IID')
bili_data = bili_snps.merge(phenotypes_bili, left_on='iid', right_on='IID').set_index('IID')
```

In [56]:
```
# save these to files
hair_data.to_csv('cleaned_data/BIN_FC2001747_data.csv')
bili_data.to_csv('cleaned_data/INI30840_data.csv')
```

In [ ]:
```

```

In [53]:
```
import pandas as pd
import numpy as np
import sys
from sklearn.decomposition import PCA
```

In [59]:
```
hair_df = pd.read_csv('cleaned_data/BIN_FC2001747_data.csv')
blackhair_df = pd.read_csv('cleaned_data/BIN_FC5001747_data.csv')
#merged = pd.merge(hair_df,blackhair_df, on = ['IID','gender'], how = 'inner')
```

In [60]:
```
red_snp_names = pd.read_csv('nam/snp_names/hair_snp_names.csv')
indices = [str(i) for i in list(red_snp_names.index)]
names = list(red_snp_names['snp'])
red_snp_names_dict = dict(zip(indices, names))
```

In [61]:
```
black_snp_names = pd.read_csv('nam/snp_names/blackhair_snp_names.csv')
indices = [str(i) for i in list(black_snp_names.index)]
names = list(black_snp_names['snp'])
black_snp_names_dict = dict(zip(indices, names))
```

In [65]:
```
hair_df.rename(columns=red_snp_names_dict, inplace=True)
```

In [66]:
```
blackhair_df.rename(columns=black_snp_names_dict, inplace=True)
```

In [74]:
```
blackhair_df.shape
```

Out[74]:
```
(80010, 1624)
```

In [75]:
```
hair_df.shape
```

Out[75]:
```
(42016, 1624)
```

In [70]:
```
overlap = set(blackhair_df.columns)&set(hair_df.columns)
```

In [77]:
```
len(list(overlap))
```

Out[77]:
```
67
```

In [ ]:
```
merged = pd.merge(hair_df,blackhair_df, on = ['gender'], how = 'inner')
```

In [ ]:
```
merged
```

In [16]:
```
bili_snp_names = pd.read_csv('nam/snp_names/bilirubin_snp_names.csv')
indices = [str(i) for i in list(bili_snp_names.index)]
names = list(bili_snp_names['snp'])
bili_snp_names_dict = dict(zip(indices, names))
```

In [17]:
```
db_snp_names = pd.read_csv('nam/snp_names/diabetes_snp_names.csv')
indices = [str(i) for i in list(db_snp_names.index)]
names = list(db_snp_names['snp'])
db_snp_names_dict = dict(zip(indices, names))
```

In [7]:
```
#hair_df = pd.read_csv('cleaned_data/BIN_FC2001747_data.csv')
bili_df = pd.read_csv('cleaned_data/INI30840_data.csv').drop('Unnamed: 0', axis = 1)
#celiac_df = pd.read_csv('cleaned_data/HC303_data.csv')
#merged_df = pd.read_csv('cleaned_data/merged.csv')#.drop('Unnamed: 0', axis = 1)
#lpa_df = pd.read_csv('cleaned_data/INI30790_data.csv')
diabetes_df = pd.read_csv('cleaned_data/INI2976_data.csv')
```

In [22]:
```
diabetes_df.rename(columns=db_snp_names_dict, inplace=True)
```

In [23]:
```
diabetes_df
```

Out[23]:

	IID	gender	rs7598922	rs6822933	rs4956041	rs35941893	rs805304	rs1043618	rs3129871	rs9273363	...	Affx-52353201	rs11078672	rs12151106	A 89021
0	1000091	0	0.0	0.0	2.0	1.0	1.0	2.0	1.0	2.0	...	1.0	0.0	1.0	
1	1000159	1	1.0	2.0	2.0	2.0	0.0	0.0	1.0	1.0	...	0.0	1.0	1.0	
2	1000278	0	1.0	0.0	2.0	1.0	0.0	0.0	1.0	0.0	...	0.0	1.0	1.0	
3	1000473	1	1.0	0.0	2.0	1.0	1.0	1.0	1.0	0.0	...	1.0	0.0	2.0	
4	1000986	0	1.0	2.0	2.0	0.0	1.0	1.0	2.0	2.0	...	1.0	1.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
25367	6025294	0	1.0	2.0	0.0	0.0	0.0	0.0	1.0	1.0	...	1.0	1.0	2.0	
25368	6025303	0	1.0	2.0	1.0	1.0	1.0	1.0	2.0	1.0	...	2.0	0.0	2.0	
25369	6025461	0	2.0	1.0	0.0	1.0	1.0	2.0	2.0	0.0	...	1.0	1.0	2.0	
25370	6026216	0	2.0	1.0	2.0	0.0	1.0	1.0	2.0	0.0	...	1.0	1.0	1.0	
25371	6026237	0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	0.0	0.0	2.0	

25372 rows × 27 columns

In [24]:
```
bili_df.rename(columns=bili_snp_names_dict, inplace=True)
bili_df
```

Out[24]:

	IID	gender	rs263526	rs6702935	rs2130621	rs12731208	rs6687430	rs11121663	rs6541010	rs6669030	...	rs911093	rs12557289	rs5907091	rs173
0	1000028	1	0.0	0.0	1.0	0.0	2.0	0.0	2.0	1.0	...	2.0	0.0	0.0	
1	1000034	0	0.0	2.0	1.0	1.0	1.0	1.0	1.0	1.0	...	2.0	0.0	2.0	
2	1000045	1	1.0	1.0	0.0	2.0	0.0	1.0	2.0	1.0	...	2.0	2.0	0.0	
3	1000052	1	0.0	0.0	2.0	1.0	2.0	1.0	1.0	2.0	...	2.0	1.0	0.0	
4	1000069	1	2.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	...	2.0	1.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
464654	6026191	1	2.0	2.0	0.0	1.0	1.0	1.0	2.0	2.0	...	2.0	1.0	1.0	
464655	6026202	0	0.0	2.0	1.0	1.0	1.0	0.0	1.0	1.0	...	2.0	2.0	0.0	
464656	6026216	0	1.0	1.0	0.0	0.0	0.0	1.0	2.0	0.0	...	2.0	2.0	0.0	
464657	6026229	0	2.0	1.0	0.0	1.0	1.0	0.0	0.0	1.0	...	2.0	0.0	0.0	
464658	6026237	0	1.0	2.0	2.0	1.0	1.0	1.0	1.0	1.0	...	2.0	2.0	0.0	

464659 rows × 1162 columns

In [37]:
```
merged = pd.merge(diabetes_df,bili_df, on = ['IID','gender'], how = 'inner')
merged
```

Out[37]:

	IID	gender	rs7598922	rs6822933	rs4956041	rs35941893	rs805304	rs1043618	rs3129871	rs9273363	...	rs911093	rs12557289	rs5907091	rs17318
0	1000091	0	0.0	0.0	2.0	1.0	1.0	2.0	1.0	2.0	...	2.0	2.0	0.0	
1	1000159	1	1.0	2.0	2.0	2.0	0.0	0.0	1.0	1.0	...	1.0	2.0	1.0	
2	1000278	0	1.0	0.0	2.0	1.0	0.0	0.0	1.0	0.0	...	2.0	2.0	0.0	
3	1000473	1	1.0	0.0	2.0	1.0	1.0	1.0	1.0	0.0	...	2.0	0.0	1.0	
4	1000986	0	1.0	2.0	2.0	0.0	1.0	1.0	2.0	2.0	...	0.0	2.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
24109	6025294	0	1.0	2.0	0.0	0.0	0.0	0.0	1.0	1.0	...	2.0	0.0	2.0	
24110	6025303	0	1.0	2.0	1.0	1.0	1.0	1.0	2.0	1.0	...	2.0	0.0	2.0	
24111	6025461	0	2.0	1.0	0.0	1.0	1.0	2.0	2.0	0.0	...	2.0	0.0	2.0	
24112	6026216	0	2.0	1.0	2.0	0.0	1.0	1.0	2.0	0.0	...	2.0	2.0	0.0	
24113	6026237	0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	2.0	2.0	0.0	

24114 rows × 1187 columns

In [5]:
```
# Add PCs
def add_pcs(df, name):
    gender_ID = df[['IID','gender']]
    phen = df[name]
    no_gender_ID_phen = df.drop(columns=['IID', 'gender',name])
    #no_gender_ID_phen = df #for merged
    pca = PCA(n_components=10)
    print('Performing PCA on SNPs ...')
    principal_components = pca.fit_transform(no_gender_ID_phen)
    print('Finished performing PCA on dataset.')
    pcs_pd = pd.DataFrame(pd.DataFrame(principal_components)).rename(columns={0:"PC0",1:"PC1",2:"PC2",3:"PC3",4:"PC4",5
    df = pd.concat([gender_ID, pcs_pd, no_gender_ID_phen, phen], axis=1)
    #df = pd.concat([pcs_pd, no_gender_ID_phen], axis=1) #sasha added
    df.to_csv('cleaned_data/{}_data_pcs.csv'.format(name))
    return df
```

In [42]:
```
# Add PCs
def add_pcs_merged(df, name1, name2):
    gender_ID = df[['IID','gender']]
    phen1 = df[name1]
    phen2 = df[name2]
    no_gender_ID_phen = df.drop(columns=['IID', 'gender',name1, name2])
    #no_gender_ID_phen = df #for merged
    pca = PCA(n_components=10)
    print('Performing PCA on SNPs ...')
    principal_components = pca.fit_transform(no_gender_ID_phen)
    print('Finished performing PCA on dataset.')
    pcs_pd = pd.DataFrame(pd.DataFrame(principal_components)).rename(columns={0:"PC0",1:"PC1",2:"PC2",3:"PC3",4:"PC4",5
    df = pd.concat([gender_ID, pcs_pd, no_gender_ID_phen, phen1, phen2], axis=1)
    #df = pd.concat([pcs_pd, no_gender_ID_phen], axis=1) #sasha added
    df.to_csv('cleaned_data/{}_{}_data_pcs.csv'.format(name1, name2))
    return df
```

In [43]:
```
add_pcs_merged(merged, 'INI2976', 'INI30840')
```

```
Performing PCA on SNPs ...
Finished performing PCA on dataset.
```

Out[43]:

	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	PC7	...	rs12557289	rs5907091	rs17318896	rs592505
0	1000091	0	-1.602856	-1.293855	-0.969427	-1.340912	2.120269	1.766469	-1.642767	0.559497	...	2.0	0.0	0.0	2.(
1	1000159	1	-1.812481	-0.918722	-1.617273	-1.639746	0.422631	-0.340432	0.341719	-1.954858	...	2.0	1.0	0.0	0.(
2	1000278	0	-0.895834	0.396114	0.813137	0.371944	1.672308	-1.608972	-2.614227	-1.113107	...	2.0	0.0	2.0	2.(
3	1000473	1	-1.545727	-0.018915	0.886567	0.831222	-0.462368	-0.165540	1.605912	-1.635657	...	0.0	1.0	2.0	1.(
4	1000986	0	-1.676266	-1.388563	-1.898639	-2.720104	1.711760	0.600410	0.594088	-0.366200	...	2.0	0.0	2.0	0.(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
24109	6025294	0	-1.905150	-0.175559	0.847913	4.360380	1.840405	-1.915762	-0.675771	-1.891599	...	0.0	2.0	0.0	0.(
24110	6025303	0	-1.270480	1.509129	-1.750454	0.274665	0.486183	0.410283	2.166974	-2.163273	...	0.0	2.0	0.0	2.(
24111	6025461	0	0.052345	-1.235150	-2.100377	-2.449739	1.149900	-0.608922	-0.478668	-0.223522	...	0.0	2.0	0.0	2.(
24112	6026216	0	-1.055751	-0.543875	0.634312	0.085382	-1.292792	0.540512	-0.026823	-0.039451	...	2.0	0.0	2.0	0.(
24113	6026237	0	-0.579897	-0.685384	-1.956769	-0.519820	1.989071	-1.842465	-2.281335	2.264533	...	2.0	0.0	2.0	0.(

24114 rows × 1197 columns

In [23]:
```
add_pcs(bili_df, 'INI30840')
```

```
Performing PCA on SNPs ...
Finished performing PCA on dataset.
```

Out[23]:

	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	PC7	...	1150	1151	1152	1153	1154	1155	1156
0	1000028	1	0.434751	0.646553	2.430707	-1.173956	0.224620	-0.186502	-1.163344	1.102598	...	2.0	0.0	0.0	0.0	2.0	2.0	0.0
1	1000034	0	-2.123274	-1.219834	1.461694	0.346124	-3.241478	-0.313167	1.546294	-0.508494	...	2.0	0.0	2.0	2.0	0.0	0.0	2.0
2	1000045	1	-2.607850	-1.531654	-0.721367	3.188655	-2.085170	0.331783	-0.038758	-0.072410	...	2.0	2.0	0.0	0.0	0.0	1.0	0.0
3	1000052	1	-0.941135	-2.243257	0.934420	1.258839	-2.523849	-0.235792	1.163712	-0.632148	...	2.0	1.0	0.0	0.0	0.0	1.0	1.0
4	1000069	1	0.346566	3.501397	0.091706	0.015239	-0.413675	-0.332229	-0.970568	-1.834571	...	2.0	1.0	0.0	1.0	1.0	2.0	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
464654	6026191	1	0.000514	3.238058	2.520103	0.827850	-1.724928	-2.029019	-1.187177	-0.541304	...	2.0	1.0	1.0	0.0	0.0	1.0	1.0
464655	6026202	0	-1.054427	-0.834939	-2.404472	-1.483921	0.478630	1.503337	-2.449843	-0.669096	...	2.0	2.0	0.0	2.0	2.0	0.0	0.0
464656	6026216	0	-0.490937	0.909108	0.355770	0.203357	-1.230320	-0.397809	-0.250201	-0.250265	...	2.0	2.0	0.0	2.0	0.0	2.0	0.0
464657	6026229	0	-1.664833	-1.771189	-0.737253	0.068791	-2.191291	1.812567	2.224974	2.231567	...	2.0	0.0	0.0	2.0	2.0	2.0	0.0
464658	6026237	0	-0.118202	-1.855761	-1.803817	-0.379063	1.536812	-1.964355	0.974051	1.751189	...	2.0	2.0	0.0	2.0	0.0	2.0	0.0

464659 rows × 1172 columns

In [ ]:

In [ ]:

In [ ]:

In [5]:
```
import numpy as np
import pandas as pd
import sys
import datetime
import os
import sklearn.metrics as sk_metrics
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import OneHotEncoder
from torch.utils.data import random_split
from joblib import Parallel, delayed
from sklearn.metrics import precision_recall_curve
import torch
import torch.nn as nn
import random
import tensorflow as tf
import sklearn
import matplotlib.pyplot as plt
import seaborn as sns

from nam.wrapper import NAMClassifier, MultiTaskNAMClassifier, NAMRegressor, MultiTaskNAMRegressor
from nam.trainer.losses import make_penalized_loss_func
from nam.models.saver import Checkpointer
from sklearn.metrics import mean_squared_error
import shap
import sklearn.metrics as metrics

from interpret.glassbox import ExplainableBoostingClassifier, ExplainableBoostingRegressor
from interpret import show
```

AUCs

CELIAC:

- celiac_val = 0.853
- celiac_test = 0.8517897160211116
- Operating point: 0.15250059355433454
- Sensitivity: 0.7371695178849145
- Specificity: 0.8409448818897638

BILIRUBIN:

- bilirubin_val = 10.487
- bilirubin_test = 10.366122117524952
- bilirubin r2 = 0.4518717503607067

Plotting Graphics

In [6]:
```
def get_importance(X_train, model):
        cols = X_train.columns
        modl_dict = {}
        for i in np.arange(X_train.shape[1]):
            modl = model.plot(i)
            x, y, conf = modl['x'], modl['y'], modl['conf_int']
            #metric of importance
            mean_feat = np.mean(y)
            importance = np.sum([np.abs(j - mean_feat) for j in y])
            if cols[i][:2] != 'PC':
                modl_dict[cols[i]] = [importance, i]
        return dict(sorted(modl_dict.items(), key=lambda item: item[1][0], reverse = True))
```

In [7]:
```
def barplot(sorted_dict, name, model, snp_names):
    keys=list(sorted_dict.keys())
    x = []
    y = []
    conf = []
    for idx in np.arange(10):
        modl = model.plot(sorted_dict[keys[idx]][1])
        x += [modl['x']]
        y += [modl['y']]
        conf += [modl['conf_int']]
    figure, axis = plt.subplots(5, 2, figsize = (13, 15), sharey = True)
    figure.tight_layout(pad=4.0)
    for i in np.arange(5):
        for j in np.arange(2):
            axis[i, j].bar(x[2*i + j], y[2*i + j], color = ['r', 'g', 'b'], yerr=conf[2*i + j])
            axis[i, j].set_xticks(list(x[2*i + j]))
            axis[i, j].set_xlabel("SNP Value")
            axis[i, j].set_ylabel("Output")
            if keys[2*i + j] == 'gender':
                axis[i, j].set_title('gender')
            else:
                axis[i, j].set_title(snp_names[keys[2*i + j]])
    plt.savefig('figures/' + name +'/'+ name +'.png', bbox_inches = 'tight')
```

In [8]:
```
def plot_auc(fpr, tpr, name):
    plt.title('Receiver Operating Characteristic')
    plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
    plt.legend(loc = 'lower right')
    plt.plot([0, 1], [0, 1],'r--')
    plt.xlim([0, 1])
    plt.ylim([0, 1])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.savefig('figures/' + name +'/'+ name + '_auc.png')
    plt.show()
```

In [9]:
```
def plot_pr(fpr, tpr, name):
    plt.title('Receiver Operating Characteristic')
    plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
    plt.legend(loc = 'lower right')
    plt.plot([0, 1], [0, 1],'r--')
    plt.xlim([0, 1])
    plt.ylim([0, 1])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.savefig('figures/' + name +'/'+ name + '_auc.png')
    plt.show()
```

In [42]:
```
def choose_operating_point(fpr, tpr, threshold, y_test):
    num_pos = sum(y_test)
    num_neg = len(y_test) - num_pos
    fp = fpr*num_neg
    tp = tpr*num_pos
    tn = num_neg - fp
    fn = num_pos - tp
    specificity = tn/(tn+fp)
    idx = np.argmax(tpr - fpr)
    op_point = thresholds[idx]
    sens = tpr[idx]

    spec = specificity[idx]
    return op_point, sens, spec
```

Classification Red Hair

In [11]:
```
red = pd.read_csv('~/sasha_jess/cleaned_data/BIN_FC2001747_data_pcs.csv')
red_nan = red.replace(-9, np.nan)
red = red_nan.fillna(red_nan.median())
```

In [12]:
```
X_train_red = red.iloc[:, 2:-1]
y_train_red = red.iloc[:, -1]
X_train_red, X_test_red, y_train_red, y_test_red = train_test_split(X_train_red, y_train_red, test_size=0.2)
```

In [ ]:
```

```

In [13]:
```
model = NAMClassifier(
            num_epochs=20,
            num_learners=5,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model.fit(X_train_red, y_train_red)
```

```
0%|          | 0/10 [00:00<?, ?it/s]

0%|          | 0/28 [00:00<?, ?it/s]

0%|          | 0/5 [00:00<?, ?it/s]

0%|          | 0/28 [00:00<?, ?it/s]

0%|          | 0/5 [00:00<?, ?it/s]

0%|          | 0/28 [00:00<?, ?it/s]

0%|          | 0/5 [00:00<?, ?it/s]

0%|          | 0/28 [00:00<?, ?it/s]

0%|          | 0/5 [00:00<?, ?it/s]

0%|          | 0/28 [00:00<?, ?it/s]

0%|          | 0/5 [00:00<?, ?it/s]
```

In [ ]:
```
tensorboard --logdir ~/sasha_jess/nam/output/0/logs/ --port=6006
```

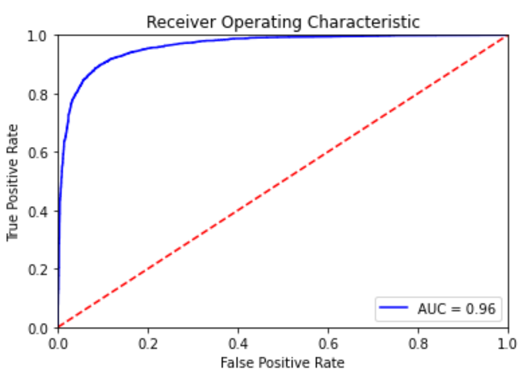
In [ ]:
```
from tensorboard import notebook
notebook.list() # View open TensorBoard instances
```

In [ ]:
```
!kill 385003
```

In [19]:
```
# calculate the fpr and tpr for all thresholds of the classification
probs = model.predict_proba(X_test_red)
preds = probs
fpr, tpr, thresholds = metrics.roc_curve(y_test_red, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.9624677822279102
```

In [16]:
```
plot_auc(fpr, tpr, 'celiac')
```



In [21]:
```
op_point, sens, spec = choose_operating_point(fpr, tpr, thresholds, y_test_red)
print('Operating point: {}'.format(op_point))
print('Sensitivity: {}'.format(sens))
print('Specificity: {}'.format(spec))
```

```
Operating point: 0.5457369010661987
Sensitivity: 0.8971783835485414
Specificity: 0.9069161534817622
```

XGboost

In [22]:
```
ebm = ExplainableBoostingClassifier(random_state=1, interactions=100)
print('fitting')
ebm.fit(X_train_red, y_train_red)
```

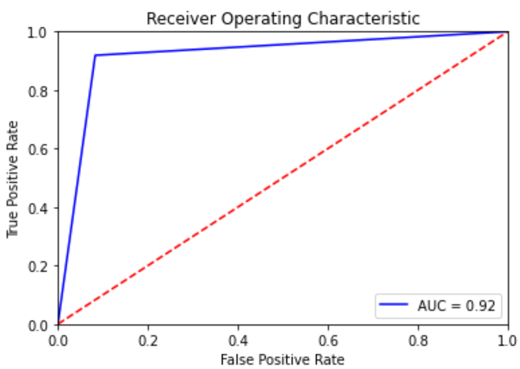
```
fitting
```

Out[22]:
```
ExplainableBoostingClassifier(interactions=100, random_state=1)
```

In [28]:
```
preds = ebm.predict(X_test_red)
preds_list = [float(i) for i in preds]
y_test_red_list = [float(i) for i in list(y_test_red.values)]
fpr, tpr, thresholds = metrics.roc_curve(y_test_red_list, preds_list)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.91801960353875
```

In [30]:
```
plot_auc(fpr, tpr, 'celiac')
```



In [43]:
```
op_point, sens, spec = choose_operating_point(fpr, tpr, thresholds, y_test_red_list)
print('Operating point: {}'.format(op_point))
print('Sensitivity: {}'.format(sens))
print('Specificity: {}'.format(spec))
```

```
Operating point: 1.0
Sensitivity: 0.9189383070301291
Specificity: 0.9171009000473709
```

Regression bilirubin

In [17]:
```
bilirubin = pd.read_csv('~/sasha_jess/cleaned_data/INI30840_data_pcs.csv')
```

In [ ]:
```
bilirubin_nan = bilirubin.replace(-9, np.nan)
bilirubin = bilirubin_nan.fillna(bilirubin_nan.median())
```

In [ ]:
```
#bilirubin.to_csv('~/sasha_jess/cleaned_data/INI30840_data.csv')
```

In [ ]:
```
bilirubin
```

In [18]:
```
X_train_bili = bilirubin.iloc[:, 2:-1]
y_train_bili = bilirubin.iloc[:, -1]
X_train_bili, X_test_bili, y_train_bili, y_test_bili = train_test_split(X_train_bili, y_train_bili, test_size=0.2)
```

In [ ]:
```
model_bili = NAMRegressor(
            num_epochs = 20,
            num_learners= 3,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            device = 'cuda',
            save_model_frequency = 5
)
model_bili.fit(X_train_bili, y_train_bili)
```

In [ ]:
```
preds = model_bili.predict(X_test_bili)
```

In [ ]:
```
sklearn.metrics.r2_score(y_test_bili, preds)
```

In [ ]:
```
sklearn.metrics.mean_squared_error(y_test_bili, preds)
```

In [ ]:
```
sorted_dict = get_importance(X_train_bili, model_bili)
```

In [ ]:
```
snp_names = pd.read_csv('snp_names/bilirubin_snp_names.csv')
indices = [str(i) for i in list(snp_names.index)]
names = list(snp_names['snp'])
snp_names_dict = dict(zip(indices, names))
```

In [ ]:
```
barplot(sorted_dict, 'bilirubin', model_bili, snp_names_dict)
```

XGBOOST

In [20]:
```
ebm = ExplainableBoostingRegressor(random_state=1, interactions=100)
print('fitting')
ebm.fit(X_train_bili, y_train_bili)
```

```
fitting

---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/tmp/ipykernel_687270/3846178656.py in <module>
      1 ebm = ExplainableBoostingRegressor(random_state=1, interactions=100)
      2 print('fitting')
----> 3 ebm.fit(X_train_bili, y_train_bili)

~/anaconda3/lib/python3.7/site-packages/interpret/glassbox/ebm/ebm.py in fit(self, X, y, sample_weight)
    549                 )
    550 
--> 551             results = provider.parallel(EBMUtils.cyclic_gradient_boost, parallel_args)
    552 
    553             # let python reclaim the dataset memory via reference counting

~/anaconda3/lib/python3.7/site-packages/interpret/provider/compute.py in parallel(self, compute_fn, compute_args_ite
r)
     18     def parallel(self, compute_fn, compute_args_iter):
     19         results = Parallel(n_jobs=self.n_jobs)(
---> 20             delayed(compute_fn)(*args) for args in compute_args_iter
     21         )
     22         return results

~/anaconda3/lib/python3.7/site-packages/joblib/parallel.py in __call__(self, iterable)
   1096 
   1097             with self._backend.retrieval_context():
-> 1098                 self.retrieve()
   1099             # Make sure that we get a last message telling us we are done
   1100             elapsed_time = time.time() - self._start_time

~/anaconda3/lib/python3.7/site-packages/joblib/parallel.py in retrieve(self)
    973             try:
    974                 if getattr(self._backend, 'supports_timeout', False):
--> 975                     self._output.extend(job.get(timeout=self.timeout))
    976                 else:
    977                     self._output.extend(job.get())

~/anaconda3/lib/python3.7/site-packages/joblib/_parallel_backends.py in wrap_future_result(future, timeout)
    565         AsyncResults.get from multiprocessing."""
    566         try:
--> 567             return future.result(timeout=timeout)
    568         except CfTimeoutError as e:
    569             raise TimeoutError from e

~/anaconda3/lib/python3.7/concurrent/futures/_base.py in result(self, timeout)
    428                 return self.__get_result()
    429 
--> 430             self._condition.wait(timeout)
    431 
    432             if self._state in [CANCELLED, CANCELLED_AND_NOTIFIED]:

~/anaconda3/lib/python3.7/threading.py in wait(self, timeout)
    294         try:    # restore state no matter what (e.g., KeyboardInterrupt)
    295             if timeout is None:
--> 296                 waiter.acquire()
    297                 gotit = True
    298             else:

KeyboardInterrupt: 
```

In [ ]:
```
preds = ebm.predict(X_test_bili)
preds_list = [float(i) for i in preds]
y_test_bili_list = [float(i) for i in list(y_test_biili.values)]
sklearn.metrics.mean_squared_error(y_test_bili_list, preds_list)
```

Classification Celiac Disease

In [22]:
```
celiac = pd.read_csv('~/sasha_jess/cleaned_data/HC303_data_pcs.csv')
celiac_nan = celiac.replace(-9, np.nan)
celiac = celiac_nan.fillna(celiac_nan.median())
```

In [23]:
```
X_train_celiac = celiac.iloc[:, 2:-1]
y_train_celiac = celiac.iloc[:, -1]
X_train_celiac, X_test_celiac, y_train_celiac, y_test_celiac = train_test_split(X_train_celiac, y_train_celiac, test_s
```

In [ ]:
```
model_celiac = NAMClassifier(
            num_epochs=50,
            num_learners=10,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_celiac.fit(X_train_celiac, y_train_celiac)
```

In [ ]:
```
import sklearn.metrics as metrics
# calculate the fpr and tpr for all thresholds of the classification
preds = model_celiac.predict_proba(X_test_celiac)
fpr, tpr, threshold = metrics.roc_curve(y_test_celiac, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

In [ ]:
```
plot_auc(fpr, tpr, 'celiac')
```

In [ ]:
```
op_point, sens, spec = choose_operating_point(fpr, tpr, thresholds, y_test_celiac)
print('Operating point: {}'.format(op_point))
print('Sensitivity: {}'.format(sens))
print('Specificity: {}'.format(spec))
```

In [ ]:
```
sorted_dict = get_importance(X_train_celiac, model_celiac)
```

In [ ]:
```
barplot(sorted_dict, 'celiac')
```

XGBOOST

In [24]:
```
ebm = ExplainableBoostingClassifier(random_state=1, interactions=100)
print('fitting')
ebm.fit(X_train_celiac, y_train_celiac)
```

```
fitting
```

Out[24]:
```
ExplainableBoostingClassifier(interactions=100, random_state=1)
```

In [26]:
```
preds = ebm.predict(X_test_celiac)
preds_list = [float(i) for i in preds]
y_test_celiac_list = [float(i) for i in list(y_test_celiac.values)]
fpr, tpr, threshold = metrics.roc_curve(y_test_celiac_list, preds_list)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.7646446486655953
```

Multitask Classification

In [45]:
```
merged_final = pd.read_csv('~/sasha_jess/cleaned_data/merged_downsampling_data_pcs.csv').drop('Unnamed: 0', axis = 1)
merged_final_nan = merged_final.replace(-9, np.nan)
merged_final = merged_final_nan.fillna(merged_final_nan.median())
X_train = merged_final.drop(['HC303',  'BIN_FC2001747'], axis = 1)
y_train = merged_final[['HC303', 'BIN_FC2001747']]
```

In [ ]:

In [46]:
```
X_train_merged, X_test_merged, y_train_merged, y_test_merged = \
train_test_split(X_train, y_train, test_size=0.2)
```

In [48]:
```
model_merged = MultiTaskNAMClassifier(
            num_epochs=50,
            num_learners=3,
            early_stop_mode='max',
            num_subnets=2,
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_merged.fit(X_train_merged, y_train_merged)
```

```
0%|          | 0/50 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]

0%|          | 0/1 [00:00<?, ?it/s]
```

In [49]:
```
pred = model_merged.predict_proba(X_test_merged)
```

In [ ]:

In [51]:
```
y_test_mtl = y_test_merged
y_test_mtl_flat = y_test_mtl.to_numpy().reshape(-1)
pred_flat = pred.reshape(-1)

non_nan_indices = y_test_mtl_flat == y_test_mtl_flat
y_test_mtl_flat = y_test_mtl_flat[non_nan_indices]
pred_flat = pred_flat[non_nan_indices]
```

In [ ]:

In [52]:
```
sk_metrics.roc_auc_score(y_test_mtl_flat, pred_flat)
```

Out[52]: 0.6766169154228855

In [ ]:
```
fpr, tpr, threshold = metrics.roc_curve(y_test_mtl_flat, pred_flat)
```

Lipoprotein A Regression

In [ ]:
```
lpa = pd.read_csv('~/sasha_jess/cleaned_data/INI30790_data_pcs.csv')
```

In [ ]:
```
lpa
```

In [ ]:
```
X_train_lpa
```

In [ ]:
```
X_train_lpa['8298']
```

In [ ]:
```
X_train_lpa = lpa.iloc[:, 2:-1]
y_train_lpa = lpa.iloc[:, -1]
X_train_lpa, X_test_lpa, y_train_lpa, y_test_lpa = train_test_split(X_train_lpa, y_train_lpa, test_size=0.2)
```

In [ ]:
```
model_lpa = NAMRegressor(
            num_epochs = 10,
            num_learners= 1,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            device = 'cuda',
            save_model_frequency = 5
)
model_lpa.fit(X_train_lpa, y_train_lpa)
```

In [ ]:
```
y_pred_lpa = model_lpa.predict(X_test_lpa)
```

In [ ]:
```
y_pred_lpa
```

In [ ]:
```
y_test_lpa
```

In [ ]:
```
sklearn.metrics.r2_score(y_test_lpa, y_pred_lpa)
```

In [ ]:
```
feature_predictions = get_feature_predictions(model_lpa, unique_features)
```

In [ ]:
```
```

Multitask Regression

In [2]:
```
merged = pd.read_csv('~/sasha_jess/cleaned_data/INI2976_INI30840_data_pcs.csv').drop('Unnamed: 0', axis = 1)
X_train = merged.drop(['INI2976',  'INI30840'], axis = 1)
y_train = merged[['INI2976', 'INI30840']]
```

In [3]:
```
X_train_merged, X_test_merged, y_train_merged, y_test_merged = \
train_test_split(X_train, y_train, test_size=0.2)
```

In [11]:
```
model_merged = MultiTaskNAMRegressor(
            num_epochs = 20,
            num_learners= 1,
            early_stop_mode='min',
            batch_size = 512,
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            num_subnets=4,
            device = 'cuda',
            save_model_frequency = 5
        )

model_merged.fit(X_train_merged, y_train_merged)
```

```
  0%|          | 0/20 [00:00<?, ?it/s]

  0%|          | 0/33 [00:00<?, ?it/s]

  0%|          | 0/6 [00:00<?, ?it/s]

  0%|          | 0/33 [00:00<?, ?it/s]

  0%|          | 0/6 [00:00<?, ?it/s]

  0%|          | 0/33 [00:00<?, ?it/s]

  0%|          | 0/6 [00:00<?, ?it/s]

  0%|          | 0/33 [00:00<?, ?it/s]

---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/tmp/ipykernel_687270/1512380651.py in <module>
     12         )
     13
```

DIABETES

In [27]:
```
diabetes = pd.read_csv('~/sasha_jess/cleaned_data/INI2976_data_pcs.csv')
diabetes_nan = diabetes.replace(-9, np.nan)
diabetes = diabetes_nan.fillna(celiac_nan.median())
```

In [28]:
```
X_train_diabetes = diabetes.iloc[:, 2:-1]
y_train_diabetes = diabetes.iloc[:, -1]
X_train_diabetes, X_test_diabetes, y_train_diabetes, y_test_diabetes = train_test_split(X_train_diabetes, y_train_diab
```

In [ ]:
```
model_diabetes = NAMClassifier(
            num_epochs=20,
            num_learners=10,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_diabetes.fit(X_train_diabetes, y_train_diabetes)
```

In [34]:
```
ebm = ExplainableBoostingClassifier(random_state=1, interactions= 10)
print('fitting')
ebm.fit(X_train_diabetes, y_train_diabetes)
```

```
fitting

Detected multiclass problem. Forcing interactions to 0. Multiclass interactions work except for global visualization
s, so the line below setting interactions to zero can be disabled if you know what you are doing.
```

Out[34]:
```
ExplainableBoostingClassifier(random_state=1)
```

In [36]:
```
preds = ebm.predict(X_test_diabetes)
preds_list = [float(i) for i in preds]
y_test_diabetes_list = [float(i) for i in list(y_test_diabetes.values)]
sklearn.metrics.mean_squared_error(y_test_diabetes_list, preds_list)
```

Out[36]:
```
205.72064039408866
```

TOTAL RESULTS

In [ ]:
```
%load_ext tensorboard
```

In [ ]:

Model without PCA: AUC RED HAIR = 0.968018375610159 MSE BILIRUBIN = 10.557112893903545 AUROC CELIAC = 0.859

Model with PCA: AUC RED HAIR = 0.954682448362645 MSE BILIRUBIN = 14.351252695419824 AUC CELIAC = 0.856

In [ ]:

In [ ]:

In [119]:
```
import numpy as np
import pandas as pd
import sys
import datetime
import os
import sklearn.metrics as sk_metrics
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import OneHotEncoder
from torch.utils.data import random_split
from joblib import Parallel, delayed
import torch
import torch.nn as nn
import random
import tensorflow as tf
import sklearn

from nam.wrapper import NAMClassifier, MultiTaskNAMClassifier, NAMRegressor
from nam.trainer.losses import make_penalized_loss_func
from nam.models.saver import Checkpointer
from sklearn.metrics import mean_squared_error
import shap
import sklearn.metrics as metrics
import matplotlib.pyplot as plt

from interpret.glassbox import ExplainableBoostingClassifier, ExplainableBoostingRegressor
from interpret import show
import seaborn as sns
```

Visualization Functions

Classification Red Hair

In [150]:
```
red = pd.read_csv('~/sasha_jess/cleaned_data/BIN_FC2001747_data_pcs.csv')
```

In [205]:
```
X_train_red = red.iloc[:, 2:-1]
y_train_red = red.iloc[:, -1]
X_train_red, X_test_red, y_train_red, y_test_red = train_test_split(X_train_red, y_train_red, test_size=0.2)
```

In [ ]:
```
model = NAMClassifier(
            num_epochs=10,
            num_learners=1,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model.fit(X_train_red, y_train_red)
```

In [ ]:
```
tensorboard --logdir ~/sasha_jess/nam/output/0/logs/ --port=6006
```

In [ ]:
```
from tensorboard import notebook
notebook.list() # View open TensorBoard instances
```

In [ ]:
```
!kill 385003
```

In [ ]:
```
# calculate the fpr and tpr for all thresholds of the classification
probs = model.predict_proba(X_test_red)
preds = probs
fpr, tpr, threshold = metrics.roc_curve(y_test_red, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

In [ ]:
```
import matplotlib.pyplot as plt
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

In [ ]:
```
train_pred = model.predict_proba(X_train)
sk_metrics.roc_auc_score(y_train, train_pred)
```

XGboost

In [172]:
```
ebm = ExplainableBoostingClassifier(random_state=1, interactions=100)
print('fitting')
ebm.fit(X_train_red, y_train_red)
#print('explain_global')
#ebm_global = ebm.explain_global()
#show([ebm_local])
#print('explain_local')
#ebm_local = ebm.explain_local(X_test_red[:5], y_test_red[:5])
#show(ebm_local)
```

```
fitting
explain_global
explain_local
```

In [207]:
```
preds = ebm.predict(X_test_red)
preds_list = [float(i) for i in preds]
y_test_red_list = [float(i) for i in list(y_test_red.values)]
fpr, tpr, threshold = metrics.roc_curve(y_test_red_list, preds_list)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.8823698590739143
```

Regression bilirubin

In [208]:
```
bilirubin = pd.read_csv('~/sasha_jess/cleaned_data/INI30840_data_pcs.csv')
```

In [209]:
```
bilirubin
```

Out[209]:

	Unnamed: 0	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	...	1150	1151	1152	1153	1154	1155	1156
0	0	1000028	1	-2.485030	-2.348752	0.184510	-0.962211	0.455937	-0.707493	0.137583	...	2	0	0	0	2	2	0
1	1	1000034	0	1.800760	-0.023756	1.154671	-5.903447	-4.150231	-3.439238	0.409306	...	2	0	2	2	0	0	2
2	2	1000045	1	-3.109698	-0.856202	0.916762	-1.732960	-1.397590	-1.488071	-1.097751	...	2	2	0	0	0	1	0
3	3	1000052	1	-2.414065	-3.122029	-0.091808	-0.446762	-1.575221	-0.659209	-0.448340	...	2	1	0	0	0	1	1
4	4	1000069	1	-2.837960	-0.789555	-0.377559	-1.281123	-0.353292	-0.966470	0.338323	...	2	1	0	1	1	2	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
464654	464654	6026191	1	-2.440049	-1.704182	-0.418708	-0.887361	0.085829	1.276581	1.648900	...	2	1	1	0	0	1	1
464655	464655	6026202	0	-2.110866	-1.129237	0.271610	0.405434	-2.280313	0.024605	-0.682216	...	2	2	0	2	2	0	0
464656	464656	6026216	0	-2.191713	-1.761692	-0.031251	-0.292418	-0.343458	0.429890	-1.262410	...	2	2	0	2	0	2	0
464657	464657	6026229	0	3.013468	0.518122	0.995146	-3.333260	-1.198200	0.624970	-0.768696	...	2	0	0	2	2	2	0
464658	464658	6026237	0	-2.910436	-1.716673	0.874340	-0.065841	-1.360964	0.828777	-0.326391	...	2	2	0	2	0	2	0

464659 rows × 1173 columns

In [210]:
```
X_train = bilirubin.iloc[:, 2:-1]
y_train = bilirubin.iloc[:, -1]
X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.2)
```

In [32]:
```
X_train
```

Out[32]:

	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	...	1149	1150	1151	1152	1153	1154	1155
250288	0	-2.257177	-1.915979	-0.135053	-1.926231	-1.697188	-0.121114	-1.351901	-1.330951	-1.184743	...	2	2	2	0	2	2	2
408495	1	-2.973367	-1.517271	0.487151	0.079645	0.132685	0.447594	-0.916410	-1.124850	0.491927	...	0	2	1	0	2	1	1
187335	0	-1.700196	-2.173533	-0.006958	1.030228	-0.719242	0.692216	0.462631	-0.567544	0.041144	...	0	0	0	2	0	0	0
334614	1	-1.789574	-1.169471	-1.151624	-0.899497	-4.682471	-1.790891	2.273037	5.923027	-1.204828	...	1	2	0	1	1	2	1
388966	1	-1.056507	2.372173	-0.263291	-0.144280	2.676570	2.266691	-2.831393	1.448969	-0.749148	...	0	2	2	0	0	0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
30463	1	-3.321366	-2.328082	0.219676	-1.036704	-0.179665	0.691796	1.137589	0.109696	-0.928056	...	0	2	1	0	1	1	0
167022	1	-1.983935	-2.291741	-0.226195	0.079009	-0.881146	0.379869	-1.089202	-1.434880	-0.978276	...	1	2	1	0	0	0	1
385762	0	-2.044293	-1.295271	0.440173	-0.430983	0.549658	2.099828	-0.881458	0.138139	-0.416995	...	-9	0	0	2	0	0	2
295876	0	-2.770462	-1.916517	-1.653349	1.309523	-2.285963	0.143431	-0.258326	-0.515810	0.240069	...	2	2	0	0	2	2	2
224277	1	-2.122917	-1.848319	0.409989	-0.242263	-1.532527	-0.606206	-0.279188	-0.684433	0.483740	...	0	0	2	1	1	1	1

371727 rows × 1170 columns

In [34]:
```
model_reg = NAMRegressor(
            num_epochs = 15,
            num_learners= 1,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            device = 'cuda',
            save_model_frequency = 5
)
model_reg.fit(X_train, y_train)
```

```
  0%|          | 0/15 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]

  0%|          | 0/309 [00:00<?, ?it/s]

  0%|          | 0/55 [00:00<?, ?it/s]
```

Out[34]:
```
<nam.wrapper.wrapper.NAMRegressor at 0x7f01af056b50>
```

In [35]:
```
y_pred = model_reg.predict(X_test)
```

In [36]:
```
sklearn.metrics.r2_score(y_test, y_pred)
```

Out[36]:
```
0.4263024664030248
```

In [41]:
```
sklearn.metrics.mean_squared_error(y_test, y_pred)
```

Out[41]:
```
11.075711010291
```

In [37]:
```
df = pd.DataFrame(columns = ['mean', 'std', 'name'])
```

In [38]:
```
#map_location=torch.device('cpu')
model=torch.load('output/0/ckpts/model-15.pt', map_location=torch.device('cpu'))
```

In [39]:
```
print("Model's state_dict:")
for param_tensor in model['model_state_dict']:
    name = param_tensor
    mean = torch.mean(torch.abs(model['model_state_dict'][param_tensor]))
    std = torch.std(model['model_state_dict'][param_tensor])
    df.loc[len(df.index)] = [mean, std, name]
```

```
Model's state_dict:
```

In [40]:
```
df.sort_values("mean", ascending=False)[:10]
```

Out[40]:

	mean	std	name
9642	tensor(1.6894)	tensor(nan)	feature_nns.267.feature_nns.4.model.0.bias
14258	tensor(1.6707)	tensor(nan)	feature_nns.396.feature_nns.0.model.0.bias
14841	tensor(1.5343)	tensor(nan)	feature_nns.412.feature_nns.1.model.0.bias
11291	tensor(1.5164)	tensor(nan)	feature_nns.313.feature_nns.3.model.0.bias
6950	tensor(1.4999)	tensor(nan)	feature_nns.193.feature_nns.0.model.0.bias
6993	tensor(1.4493)	tensor(nan)	feature_nns.194.feature_nns.1.model.0.bias
7994	tensor(1.4479)	tensor(nan)	feature_nns.222.feature_nns.0.model.0.bias
13545	tensor(1.4112)	tensor(nan)	feature_nns.376.feature_nns.1.model.0.bias
1283	tensor(1.3761)	tensor(nan)	feature_nns.35.feature_nns.3.model.0.bias
9693	tensor(1.3491)	tensor(nan)	feature_nns.269.feature_nns.1.model.0.bias

Classification Celiac Disease

In [97]:
```
celiac = pd.read_csv('~/sasha_jess/cleaned_data/HC303_data_pcs.csv')
```

In [98]:
```
X_train_celiac = celiac.iloc[:, 2:-1]
y_train_celiac = celiac.iloc[:, -1]
X_train_celiac, X_test_celiac, y_train_celiac, y_test_celiac = train_test_split(X_train_celiac, y_train_celiac, test_si
```

In [99]:
```
celiac.shape
```

Out[99]:
```
(6388, 437)
```

In [108]:
```
celiac_nan = celiac.replace(-9, np.nan)
```

In [110]:
```
celiac_no9 = celiac_nan.dropna()
X_train_celiac_no9 = celiac_no9.iloc[:, 2:-1]
y_train_celiac_no9 = celiac_no9.iloc[:, -1]
X_train_celiac_no9, X_test_celiac_no9, y_train_celiac_no9, y_test_celiac_no9 = train_test_split(X_train_celiac_no9, y_t
```

In [ ]:

In [112]:
```
model_celiac = NAMClassifier(
            num_epochs=60,
            num_learners=1,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_celiac.fit(X_train_celiac_no9, y_train_celiac_no9)
```

```
  0%|          | 0/60 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]
```

In [113]:
```
preds = model_celiac.predict_proba(X_test_celiac_no9)
fpr, tpr, threshold = metrics.roc_curve(y_test_celiac_no9, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.8153405902816743
```

In [115]:
```
celiac_median = celiac_nan.fillna(celiac_nan.median())
celiac_median
```

Out[115]:

	Unnamed: 0	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	...	800609	800810	801676	801897	802098	80:
0	0	1000370	1	-1.217298	-2.465534	9.275654	0.473448	-0.495340	-1.355526	2.292859	...	0.0	1.0	1.0	1.0	1.0	
1	1	1000998	1	5.831720	1.449414	0.872061	0.034517	3.357673	5.772951	-0.004874	...	1.0	1.0	1.0	0.0	1.0	
2	2	1001904	1	-8.586012	13.038083	-2.483975	3.420657	-3.249482	3.732272	-1.205457	...	0.0	1.0	1.0	1.0	1.0	
3	3	1001962	1	-0.694355	-1.732359	-1.275282	0.207636	-1.121330	0.782068	-0.299743	...	2.0	0.0	1.0	1.0	2.0	
4	4	1002535	1	1.810307	-0.264882	0.214023	0.326418	0.402879	-3.676624	10.926567	...	2.0	2.0	1.0	2.0	1.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
6383	6383	6019403	1	1.248860	0.037597	9.414793	-0.107599	-1.597313	-4.257469	-1.182120	...	0.0	2.0	2.0	1.0	1.0	
6384	6384	6021238	0	-2.291755	0.653849	-1.723715	4.957746	-5.634397	4.913494	-0.232687	...	0.0	0.0	0.0	0.0	2.0	
6385	6385	6021441	0	0.308161	-1.886748	-1.930490	-1.820300	1.437775	-1.208758	7.016675	...	2.0	0.0	2.0	2.0	0.0	
6386	6386	6022243	1	-1.599610	-0.878350	-1.405513	0.178063	-0.505243	0.460698	-0.154089	...	1.0	1.0	1.0	1.0	1.0	
6387	6387	6022298	1	-0.825470	-0.676547	-1.695263	0.540122	0.232044	-0.862867	1.204643	...	0.0	0.0	0.0	2.0	1.0	

6388 rows × 437 columns

In [116]:
```
X_train_celiac_median = celiac_median.iloc[:, 2:-1]
y_train_celiac_median = celiac_median.iloc[:, -1]
X_train_celiac_median, X_test_celiac_median, y_train_celiac_median, y_test_celiac_median = train_test_split(X_train_cel
```

In [117]:
```
model_celiac = NAMClassifier(
            num_epochs=60,
            num_learners=1,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_celiac.fit(X_train_celiac_median, y_train_celiac_median)
```

```
  0%|          | 0/60 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]
```

In [118]:
```
preds = model_celiac.predict_proba(X_test_celiac_median)
fpr, tpr, threshold = metrics.roc_curve(y_test_celiac_median, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.8394068159831499
```

In [5]:
```
model_celiac = NAMClassifier(
            num_epochs=10,
            num_learners=1,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_celiac.fit(X_train_celiac, y_train_celiac)
```

```
  0%|          | 0/10 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]
```

Out[5]:
```
<nam.wrapper.wrapper.NAMClassifier at 0x7f340e0f7650>
```

In [5]:
```
import sklearn.metrics as metrics
# calculate the fpr and tpr for all thresholds of the classification
preds = model_celiac.predict_proba(X_test_celiac)
fpr, tpr, threshold = metrics.roc_curve(y_test_celiac, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.8090402264345141
```

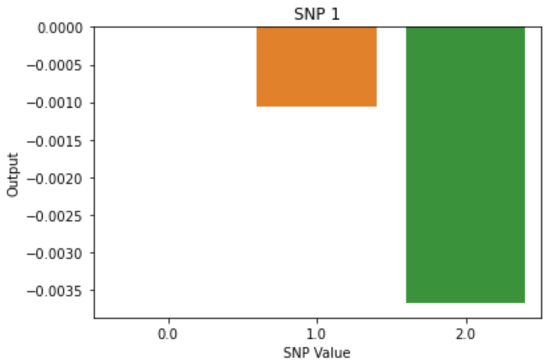
In [8]:
```
unique_features = compute_features(X_train_celiac)
```

In [13]:
```
unique_features[0:2]
```

Out[13]:
```
[array([[0.],
        [1.]]),
 array([[-11.40161781],
        [-10.4719631 ],
        [-10.38290237],
        ...,
        [ 18.30751206],
        [ 18.36454845],
        [ 18.39043968]])]
```

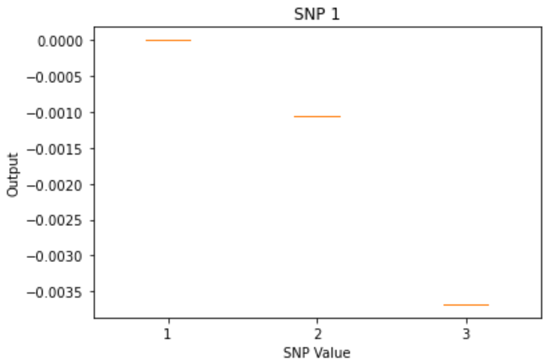
In [89]:
```
def barplot(model, feature_index, feature_name):

    df = pd.DataFrame({'x':values['x'], 'y':values['y']})
    sns.barplot(data=df, x='x', y='y')
    plt.xlabel("SNP Value")
    plt.ylabel("Output")
    plt.title(feature_name)
```

In [90]:
```
barplot(val, "SNP 1")
```



In [94]:
```
def boxplot(feature_index, feature_name):
    plt.boxplot([[values['y'][i]] for i in range(len(values['y']))])
    plt.xlabel("SNP Value")
    plt.ylabel("Output")
    plt.title(feature_name)
```

In [95]:
```
boxplot(val, "SNP 1")
```



In [ ]:
```
fig, axs = plt.subplots(5, 5)
axs[0, 0].plot(x, y)
axs[0, 0].set_title('Axis [0, 0]')
axs[0, 1].plot(x, y, 'tab:orange')
axs[0, 1].set_title('Axis [0, 1]')
axs[1, 0].plot(x, -y, 'tab:green')
axs[1, 0].set_title('Axis [1, 0]')
axs[1, 1].plot(x, -y, 'tab:red')
axs[1, 1].set_title('Axis [1, 1]')
```

In [25]:
```
model_celiac.feature_nns[0](array([[0.],[1.]]), training=nn_model._false)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
/tmp/ipykernel_680417/2389289408.py in <module>
----> 1 model_celiac.feature_nns[0](array([[0.],[1.]]), training=nn_model._false)

AttributeError: 'NAMClassifier' object has no attribute 'feature_nns'
```

In [15]:
```
feature_predictions
```

Out[15]:
```
[array([], dtype=float64), array([], dtype=float64)]
```

Multitask Classification

In [3]:
```
merged_final = pd.read_csv('~/sasha_jess/cleaned_data/merged_data_pcs.csv').drop('Unnamed: 0', axis = 1)
X_train = merged_final.drop(['HC303',  'BIN_FC2001747'], axis = 1)
y_train = merged_final[['HC303', 'BIN_FC2001747']]
```

In [4]:
```
X_train_merged, X_test_merged, y_train_merged, y_test_merged = \
train_test_split(X_train, y_train, test_size=0.2)
```

In [11]:
```
model_merged = MultiTaskNAMClassifier(
            num_epochs=10,
            num_learners=1,
            early_stop_mode='max',
            num_subnets=1,
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_merged.fit(X_train_merged, y_train_merged)
```

```
  0%|          | 0/10 [00:00<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:00<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:00<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:00<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:00<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:00<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:01<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:01<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:01<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:01<?, ?it/s]

  0%|          | 0/324 [00:00<?, ?it/s]

  0%|          | 0/58 [00:01<?, ?it/s]
```

Out[11]:
```
<nam.wrapper.wrapper.MultiTaskNAMClassifier at 0x7fd254516790>
```

In [17]:
```
pred = model_merged.predict_proba(X_test_merged)
```

In [18]:
```
y_test_mtl = y_test_merged
y_test_mtl_flat = y_test_mtl.to_numpy().reshape(-1)
pred_flat = pred.reshape(-1)

non_nan_indices = y_test_mtl_flat == y_test_mtl_flat
y_test_mtl_flat = y_test_mtl_flat[non_nan_indices]
pred_flat = pred_flat[non_nan_indices]
```

In [19]:
```
sk_metrics.roc_auc_score(y_test_mtl_flat, pred_flat)
```

Out[19]:
```
0.9413086871193995
```

In [53]:
```
fpr, tpr, threshold = metrics.roc_curve(y_test_mtl_flat, pred_flat)
```

Out[53]:
```
(array([0.        , 0.        , 0.        , ..., 0.99873057, 0.99873057,
        1.        ]),
 array([0.00000000e+00, 2.08724692e-04, 3.54831977e-03, ...,
        9.99791275e-01, 1.00000000e+00, 1.00000000e+00]))
```

Lipoprotein A Regression

In [15]:
```
lpa = pd.read_csv('~/sasha_jess/cleaned_data/INI30790_data_pcs.csv')
```

In [42]:
```
lpa
```

Out[42]:

	Unnamed: 0	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	...	8299	8300	8301	8302	8303	8304	8305
0	0	1000034	0	-8.463668	-4.503988	-2.394768	-1.174971	-0.044680	-1.402468	-0.170921	...	1	0	1	1	1	0	0
1	1	1000045	1	-8.516355	-3.342029	-2.112510	-0.203010	-0.307320	-1.382747	-0.005861	...	2	0	1	2	0	2	1
2	2	1000052	1	-8.453097	-1.724751	-1.256663	1.161195	-0.476603	-0.607844	-1.237636	...	0	0	2	1	0	0	0
3	3	1000069	1	-7.790867	-4.254623	-1.169019	-0.250887	0.593779	-1.362112	0.701024	...	2	0	2	0	0	0	0
4	4	1000076	0	-6.938971	-3.263756	-2.142954	-0.660479	1.536158	0.167482	1.655789	...	2	0	2	1	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
374216	374216	6026191	1	-3.121949	-3.322450	-1.350404	-2.021502	1.416659	0.114344	0.700689	...	1	0	2	0	0	1	0
374217	374217	6026202	0	-8.183362	-3.225504	-1.576846	-0.113021	-1.511518	-1.104854	0.425476	...	1	0	2	1	0	1	1
374218	374218	6026216	0	-8.723990	-2.373335	-2.961949	-1.069825	-1.241048	-1.802067	-0.427068	...	0	1	1	1	0	1	1
374219	374219	6026229	0	10.319877	1.142049	0.827265	7.337850	0.840700	-3.773776	-13.840719	...	1	1	2	0	0	0	0
374220	374220	6026237	0	-7.867263	-2.749188	-3.392081	2.290059	-0.246871	-0.693504	-1.062847	...	1	0	2	0	0	1	0

374221 rows × 8322 columns

In [44]:
```
X_train_lpa
```

Out[44]:

	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	...	8298	8299	8300	8301	8302	8303
132459	0	-6.023809	5.052635	5.027735	-4.362239	-8.278340	1.334963	11.779741	-1.057063	-2.993320	...	1	0	0	1	0	1
345312	1	-8.099240	-3.150713	-1.505931	-0.012974	-0.550949	-1.236086	-0.372476	1.077959	-2.070570	...	0	2	0	1	1	1
324511	0	-7.053272	2.141121	-0.275334	-1.625025	-0.933975	-2.364370	-0.715473	-0.909299	-4.742753	...	0	1	0	1	1	0
303570	1	7.388969	-13.978446	18.560159	21.990548	-4.680849	-3.883795	8.765657	3.547562	-4.137051	...	2	2	0	2	0	2
37438	1	-7.411431	-3.707891	0.269293	1.227772	0.645862	7.537215	-0.990561	0.009743	-3.223438	...	2	2	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
259178	0	-9.067180	-2.829867	-0.541727	-0.403064	-0.718799	-2.472490	-0.046392	1.579286	-2.206102	...	1	1	1	2	1	0
365838	0	6.664371	-8.006310	-6.629185	-11.103753	3.245942	-1.872478	-1.069954	-7.328004	-4.334372	...	1	0	0	2	1	0
131932	1	-5.218249	13.820721	9.213888	-10.840541	-5.735231	-1.059672	8.463401	1.642056	-3.413388	...	1	2	1	1	0	0
146867	0	-1.817478	-2.521476	-1.846181	-4.688566	0.152493	-2.806698	-1.153609	-0.350275	-4.464217	...	2	0	0	1	1	1
121958	1	-9.266508	-0.294465	-0.962962	-0.960761	-1.296336	-2.766436	1.558063	0.935239	-1.524030	...	1	1	0	2	1	0

299376 rows × 8319 columns

In [1]:
```
X_train_lpa['8298']
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipykernel_673147/896281468.py in <module>
----> 1 X_train_lpa['8298']

NameError: name 'X_train_lpa' is not defined
```

In [43]:
```
X_train_lpa = lpa.iloc[:, 2:-1]
y_train_lpa = lpa.iloc[:, -1]
X_train_lpa, X_test_lpa, y_train_lpa, y_test_lpa = train_test_split(X_train_lpa, y_train_lpa, test_size=0.2)
```

In [45]:
```
model_lpa = NAMRegressor(
            num_epochs = 10,
            num_learners= 1,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            device = 'cuda',
            save_model_frequency = 5
)
model_lpa.fit(X_train_lpa, y_train_lpa)
```

```
  0%|          | 0/10 [00:00<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:00<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:02<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:01<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:02<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:02<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:01<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:02<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:00<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:01<?, ?it/s]
  0%|          | 0/249 [00:00<?, ?it/s]
  0%|          | 0/44 [00:00<?, ?it/s]
```

Out[45]:
```
<nam.wrapper.wrapper.NAMRegressor at 0x7f0317fea1d0>
```

In [ ]:
```
y_pred_lpa = model_lpa.predict(X_test_lpa)
```

In [ ]:
```
y_pred_lpa
```

In [ ]:
```
y_test_lpa
```

In [ ]:
```
sklearn.metrics.r2_score(y_test_lpa, y_pred_lpa)
```

In [ ]:
```
feature_predictions = get_feature_predictions(model_lpa, unique_features)
```

In [ ]:

TOTAL RESULTS

In [129]:
```
%load_ext tensorboard
```

In [ ]:

Model without PCA: AUC RED HAIR = 0.968018375610159 MSE BILIRUBIN = 10.557112893903545 AUROC CELIAC = 0.859

Model with PCA: AUC RED HAIR = 0.954682448362645 MSE BILIRUBIN = 14.351252695419824 AUC CELIAC = 0.856

In [ ]:

In [ ]:

In [1]:
```
import numpy as np
import pandas as pd
import sys
import datetime
import os
import sklearn.metrics as sk_metrics
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import OneHotEncoder
from torch.utils.data import random_split
from joblib import Parallel, delayed
import torch
import torch.nn as nn
import random
import tensorflow as tf
import sklearn
import matplotlib.pyplot as plt
import seaborn as sns

from nam.wrapper import NAMClassifier, MultiTaskNAMClassifier, NAMRegressor, MultiTaskNAMRegressor
from nam.trainer.losses import make_penalized_loss_func
from nam.models.saver import Checkpointer
from sklearn.metrics import mean_squared_error
import shap
import sklearn.metrics as metrics

from interpret.glassbox import ExplainableBoostingClassifier, ExplainableBoostingRegressor
from interpret import show
```

```
2022-12-14 17:56:38.635128: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized
with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical opera
tions:  AVX2 AVX512F AVX512_VNNI FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-12-14 17:56:38.851924: I tensorflow/core/util/port.cc:104] oneDNN custom operations are on. You may see slightl
y different numerical results due to floating-point round-off errors from different computation orders. To turn them
off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2022-12-14 17:56:39.571187: W tensorflow/compiler/xla/stream_executor/platform/default/dso_loader.cc:64] Could not l
oad dynamic library 'libnvinfer.so.7'; dlerror: libnvinfer.so.7: cannot open shared object file: No such file or dir
ectory
2022-12-14 17:56:39.571309: W tensorflow/compiler/xla/stream_executor/platform/default/dso_loader.cc:64] Could not l
oad dynamic library 'libnvinfer_plugin.so.7'; dlerror: libnvinfer_plugin.so.7: cannot open shared object file: No su
ch file or directory
2022-12-14 17:56:39.571318: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Cannot dlopen so
me TensorRT libraries. If you would like to use Nvidia GPU with TensorRT, please make sure the missing libraries men
tioned above are installed properly.
```

Plotting Graphics

In [2]:
```
def get_importance(X_train, model):
        cols = X_train.columns
        modl_dict = {}
        for i in np.arange(X_train.shape[1]):
            modl = model.plot(i)
            x, y, conf = modl['x'], modl['y'], modl['conf_int']
            #metric of importance
            mean_feat = np.mean(y)
            importance = np.sum([np.abs(j - mean_feat) for j in y])
            if cols[i][:2] != 'PC':
                modl_dict[cols[i]] = [importance, i]
        return dict(sorted(modl_dict.items(), key=lambda item: item[1][0], reverse = True))
```

In [33]:
```
def barplot(sorted_dict, name, model, snp_names):
    keys=list(sorted_dict.keys())
    x = []
    y = []
    conf = []
    for idx in np.arange(10):
        modl = model.plot(sorted_dict[keys[idx]][1])
        x += [modl['x']]
        y += [modl['y']]
        conf += [modl['conf_int']]
    figure, axis = plt.subplots(5, 2, figsize = (13, 15), sharey = True)
    figure.tight_layout(pad=4.0)
    for i in np.arange(5):
        for j in np.arange(2):
            axis[i, j].bar(x[2*i + j], y[2*i + j], color = ['r', 'g', 'b'], yerr=conf[2*i + j])
            axis[i, j].set_xticks(list(x[2*i + j]))
            axis[i, j].set_xlabel("SNP Value")
            axis[i, j].set_ylabel("Output")
            if keys[2*i + j] == 'gender':
                axis[i, j].set_title('gender')
            else:
                axis[i, j].set_title(snp_names[keys[2*i + j]])
    plt.savefig('figures/' + name +'/'+ name +'.png', bbox_inches = 'tight')
```

In [4]:
```
def plot_auc(fpr, tpr, name):
    plt.title('Receiver Operating Characteristic')
    plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
    plt.legend(loc = 'lower right')
    plt.plot([0, 1], [0, 1],'r--')
    plt.xlim([0, 1])
    plt.ylim([0, 1])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.savefig('figures/' + name +'/'+ name + '_auc.png')
    plt.show()
```

In [5]:
```
def choose_operating_point(fpr, tpr, threshold, y_test):
    num_pos = sum(y_test)
    num_neg = len(y_test) - num_pos
    fp = fpr*num_neg
    tp = tpr*num_pos
    tn = num_neg - fp
    fn = num_pos - tp
    specificity = tn/(tn+fp)

    idx = np.argmax(tpr - fpr)
    op_point = thresholds[idx]
    sens = tpr[idx]

    spec = specificity[idx]
    return op_point, sens, spec
```

Classification Red Hair

In [5]:
```
red = pd.read_csv('~/sasha_jess/cleaned_data/BIN_FC2001747_data_pcs.csv')
red_nan = red.replace(-9, np.nan)
red = red_nan.fillna(red_nan.median())
```

In [6]:
```
red
```

Out[6]:

	Unnamed: 0	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	...	1612	1613	1614	1615	1616	1617	1618
0	0	1000211	0	-3.268254	-1.382627	-0.623126	-1.243778	-1.304246	0.220896	0.637238	...	0.0	0.0	0.0	0.0	2.0	0.0	0.0
1	1	1000278	0	1.004389	1.075150	-3.754420	2.084001	4.480223	-1.288013	2.520368	...	0.0	0.0	0.0	2.0	2.0	0.0	0.0
2	2	1000341	0	5.446088	-8.792445	18.242699	-2.523716	9.206329	-6.541364	-9.701964	...	0.0	0.0	2.0	2.0	2.0	0.0	0.0
3	3	1000636	1	8.655483	-2.670860	-5.328436	-0.371940	0.426607	0.716604	5.188667	...	1.0	0.0	0.0	1.0	0.0	0.0	1.0
4	4	1000672	1	-3.115452	-0.629264	-0.962107	-1.862865	0.086035	-1.506934	1.168469	...	1.0	0.0	0.0	0.0	1.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
42011	42011	6025649	1	-2.712707	0.613993	-0.044142	-1.229437	0.025995	-2.212934	2.033798	...	1.0	0.0	1.0	0.0	2.0	0.0	0.0
42012	42012	6025785	1	-3.977321	0.120709	1.106287	-1.277016	-0.854530	0.746672	0.095921	...	0.0	2.0	2.0	0.0	1.0	0.0	0.0
42013	42013	6026033	1	3.162652	-4.642354	1.063009	-0.143978	-2.843617	-0.673559	0.508429	...	0.0	0.0	0.0	2.0	1.0	1.0	1.0
42014	42014	6026047	1	10.288439	3.439637	-9.143045	-1.841574	6.155416	-0.582056	1.823783	...	1.0	0.0	1.0	0.0	2.0	0.0	0.0
42015	42015	6026187	0	-2.263736	-0.558840	-0.061548	-1.398253	-1.370929	0.250721	0.731687	...	0.0	0.0	0.0	0.0	2.0	0.0	0.0

42016 rows × 1635 columns

In [7]:
```
X_train_red = red.iloc[:, 2:-1]
y_train_red = red.iloc[:, -1]
X_train_red, X_test_red, y_train_red, y_test_red = train_test_split(X_train_red, y_train_red, test_size=0.2)
```

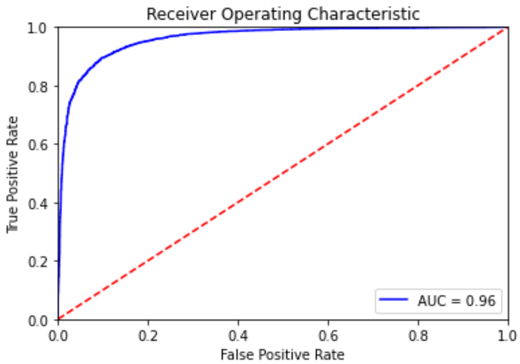
In [9]:
```
model = NAMClassifier(
            num_epochs=20,
            num_learners=5,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model.fit(X_train_red, y_train_red)
```

```
  0%|          | 0/20 [00:00<?, ?it/s]

  0%|          | 0/28 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/28 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/28 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/28 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/28 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]
```

In [20]:
```
import sklearn.metrics as metrics
# calculate the fpr and tpr for all thresholds of the classification
preds = model.predict_proba(X_test_red)
fpr, tpr, thresholds = metrics.roc_curve(y_test_red, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.9602793901446418
```

In [18]:
```
plot_auc(fpr, tpr, 'red_hair')
```



In [22]:
```
op_point, sens, spec = choose_operating_point(fpr, tpr, thresholds, y_test_red)
print('Operating point: {}'.format(op_point))
print('Sensitivity: {}'.format(sens))
print('Specificity: {}'.format(spec))
```

```
Operating point: 0.5361514000440822
Sensitivity: 0.8941005802707931
Specificity: 0.9032333645735707
```

In [51]:
```
snp_names = pd.read_csv('snp_names/hair_snp_names.csv')
indices = [str(i) for i in list(snp_names.index)]
names = list(snp_names['snp'])
snp_names_dict = dict(zip(indices, names))
snp_names_dict['1224']
```

Out[51]:
```
'Affx-35293625'
```

In [45]:
```
X_train_red.rename(columns = snp_names_dict, inplace=True)
```

In [46]:
```
X_train_red
```

Out[46]:

	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	...	rs73638243	rs9284560	rs5925408	rs5925
764	0	-5.801121	-0.101687	-1.006984	10.928892	-1.247342	-4.647845	1.919895	0.245414	-1.630605	...	2.0	0.0	0.0	
19832	0	-2.024449	-1.239940	-1.469889	-2.278172	0.864365	-0.656380	0.565291	2.966309	0.226070	...	0.0	0.0	2.0	
39736	0	0.111652	-3.868110	1.573269	-1.047966	-2.377681	0.791014	-0.852324	1.114173	2.716383	...	2.0	2.0	0.0	
18112	0	-4.002384	-0.345163	-0.118570	2.549045	0.680333	3.976411	-1.786649	-6.252633	1.809814	...	2.0	0.0	0.0	
30405	0	-1.668807	-0.104778	-1.222281	1.171282	1.880410	4.568253	-1.426221	-5.409696	3.421270	...	2.0	0.0	2.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9667	1	-3.831336	0.251540	0.184284	-2.155053	-0.974420	-0.868822	0.493124	-0.562002	0.173259	...	0.0	0.0	0.0	
1047	1	3.845043	3.384504	3.736292	2.277863	0.890305	4.201462	3.688941	-4.111348	5.484538	...	0.0	0.0	0.0	
35769	0	19.415855	11.215508	-7.370776	2.530249	1.553089	-3.105133	-8.461965	5.716940	4.567992	...	2.0	2.0	0.0	
30360	0	-3.377600	-0.924270	-0.607334	-2.018207	0.364636	0.197649	0.560174	0.075014	-1.093584	...	0.0	0.0	0.0	
39458	0	-3.481126	0.260096	-1.109342	-3.153748	0.794622	-0.531910	-0.524909	-1.489778	0.300069	...	2.0	2.0	2.0	

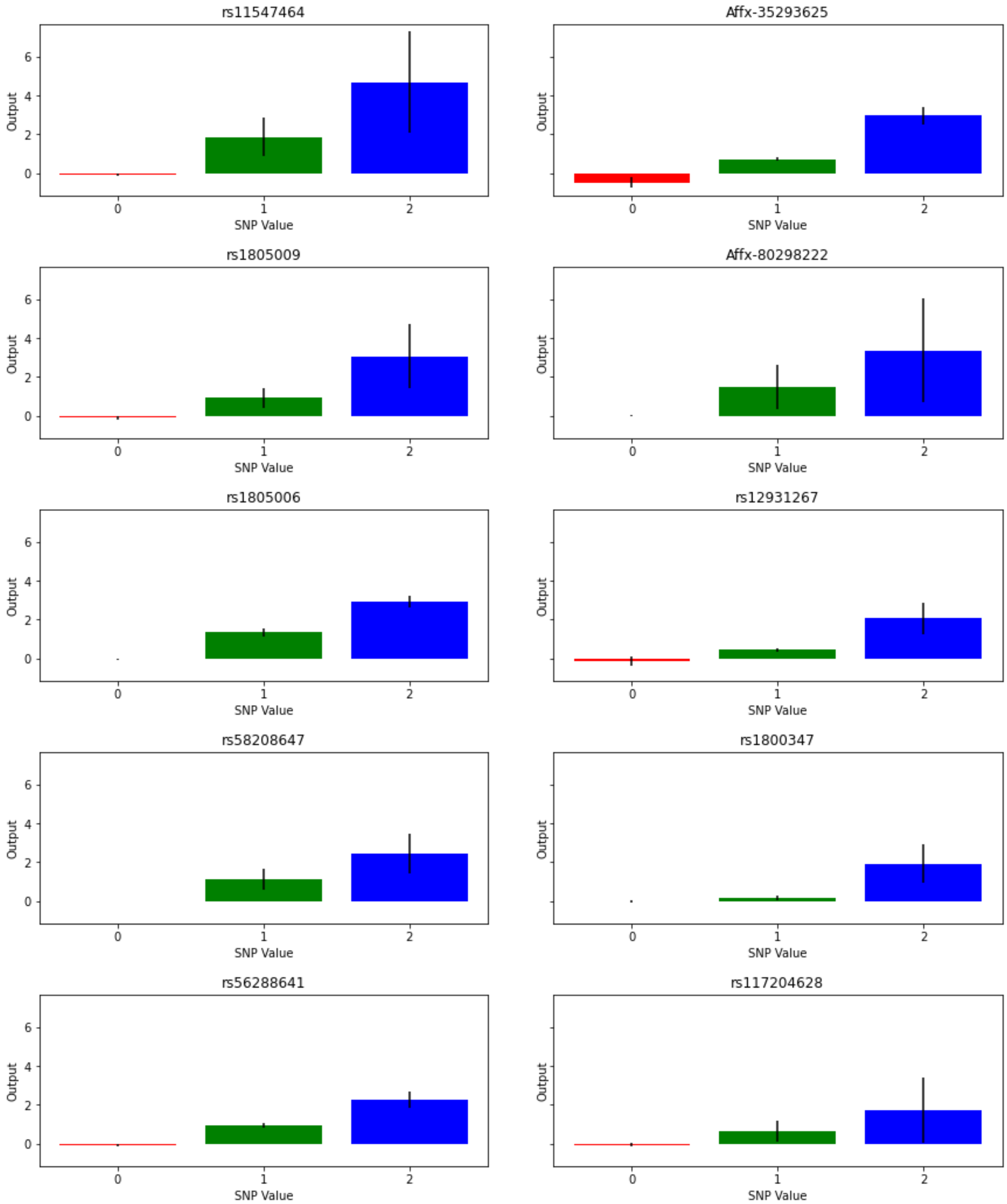
33612 rows × 1632 columns

In [26]:
```
sorted_dict = get_importance(X_train_red, model)
```

In [47]:
```
print(sorted_dict)
```


```
{'1223': [5.054896, 1234], '1224': [3.820513, 1235], '1227': [3.5428548, 1238], '1226': [3.5237305, 1237], '1222':
[3.0176663, 1233], '1211': [2.5708318, 1222], '1202': [2.5193224, 1213], '1210': [2.464094, 1221], '1207': [2.436868
7, 1218], '1234': [1.9204848, 1245], '1220': [1.8515489, 1231], '1200': [1.7427895, 1211], '1209': [1.2589726, 122
0], '1196': [1.2361197, 1207], '1214': [1.1302421, 1225], '1225': [0.8916435, 1236], '1439': [0.8270656, 1450], '121
7': [0.77163494, 1228], '1215': [0.6874274, 1226], '1221': [0.6511353, 1232], '1212': [0.64848566, 1223], '1440':
[0.63732755, 1451], '1232': [0.6015847, 1243], '1228': [0.60048664, 1239], '1230': [0.57991326, 1241], '1231': [0.57
727724, 1242], '1205': [0.57603437, 1216], '1233': [0.5489315, 1244], '1093': [0.51603127, 1104], '1219': [0.4783936
7, 1230], '1197': [0.44300425, 1208], '1237': [0.40715614, 1248], '1203': [0.38267902, 1214], '1235': [0.37854975, 1
246], '1204': [0.34671652, 1215], '1240': [0.34506863, 1251], '1433': [0.25392163, 1444], '1239': [0.25375468, 125
0], '1213': [0.25280634, 1224], '1201': [0.22434941, 1212], '470': [0.21658564, 481], '1441': [0.19561796, 1452], '1
229': [0.19027677, 1240], '460': [0.1762267, 471], '1216': [0.16476354, 1227], '1236': [0.16063231, 1247], '1436':
[0.1439797, 1447], '1435': [0.14320453, 1446], '471': [0.1388904, 482], '1442': [0.13570242, 1453], '102': [0.127364
74, 113], '1432': [0.12602971, 1443], '1206': [0.1257503, 1217], '140': [0.115018554, 151], '1430': [0.11244367, 144
1], '244': [0.105371654, 255], '517': [0.1035784, 528], '1438': [0.10285185, 1449], '1437': [0.102301426, 1448], '11
98': [0.09928207, 1209], '1161': [0.09465761, 1172], '192': [0.09288558, 203], '979': [0.08804686, 990], '488': [0.0
850621, 499], '1618': [0.08479543, 1629], '198': [0.08272663, 209], '1173': [0.08216948, 1184], '1383': [0.07864623
5, 1394], '516': [0.07666083, 527], '1434': [0.07366395, 1445], '513': [0.07247956, 524], '76': [0.06899826, 87], '1
309': [0.068106025, 1320], '75': [0.06781309, 86], '897': [0.06578267, 908], '1613': [0.06521355, 1624], 'gender':
[0.065051675, 0], '1357': [0.06455739, 1368], '939': [0.06323622, 950], '469': [0.06168385, 480], '563': [0.06103770
```

In [49]:
```
barplot(sorted_dict, 'red_hair', model, snp_names_dict)
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```
tensorboard --logdir ~/sasha_jess/nam/output/0/logs/ --port=6006
```

In [ ]:
```
from tensorboard import notebook
notebook.list() # View open TensorBoard instances
```

In [ ]:
```
!kill 385003
```

In [ ]:
```
# calculate the fpr and tpr for all thresholds of the classification
probs = model.predict_proba(X_test_red)
preds = probs
fpr, tpr, threshold = metrics.roc_curve(y_test_red, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

In [ ]:
```
import matplotlib.pyplot as plt
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

In [ ]:
```
train_pred = model.predict_proba(X_train)
sk_metrics.roc_auc_score(y_train, train_pred)
```

Diabetes Regression

In [5]:
```
diabetes = pd.read_csv('~/sasha_jess/cleaned_data/INI2976_data_pcs.csv')
diabetes.head()
```

Out[5]:

	Unnamed: 0	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	...	15	16	17	18	19	20	21	22	23	INI297
0	0	1000091	0	0.931921	-0.814756	0.816629	-0.850493	-1.025777	-0.287703	0.249742	...	1.0	0.0	1.0	0.0	1.0	1.0	2.0	2.0	2.0	26
1	1	1000159	1	-1.372248	-0.292864	-0.907618	0.730283	0.219157	-1.153402	-0.820918	...	0.0	1.0	1.0	2.0	1.0	1.0	0.0	1.0	1.0	57
2	2	1000278	0	-0.764714	0.804161	-0.070033	-1.387962	0.174131	0.189481	1.468399	...	0.0	1.0	1.0	2.0	2.0	2.0	2.0	0.0	2.0	63
3	3	1000473	1	0.210986	-0.026786	1.705204	0.778659	-0.114489	1.025245	0.587127	...	1.0	0.0	2.0	2.0	1.0	2.0	2.0	2.0	0.0	65
4	4	1000986	0	1.142438	-0.100836	0.028023	-1.519589	1.217618	-0.052774	0.665365	...	1.0	1.0	0.0	2.0	2.0	0.0	2.0	0.0	2.0	25

5 rows × 38 columns

In [6]:
```
X_train_db = diabetes.iloc[:, 2:-1]
y_train_db = diabetes.iloc[:, -1]
X_train_db, X_test_db, y_train_db, y_test_db = train_test_split(X_train_db, y_train_db, test_size=0.2)
```

In [8]:
```
y_train_db
```

Out[8]:
```
25007    58.0
24913    67.0
7335     50.0
17549    40.0
21576    66.0
         ...
5859     42.0
9026     55.5
3478     44.0
16359    44.0
14571    53.0
Name: INI2976, Length: 20297, dtype: float64
```

In [17]:
```
model_db = NAMRegressor(
            num_epochs = 60,
            num_learners= 10,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            device = 'cuda',
            save_model_frequency = 5
)
model_db.fit(X_train_db, y_train_db)
```

```
  0%|          | 0/60 [00:00<?, ?it/s]

  0%|          | 0/17 [00:00<?, ?it/s]

  0%|          | 0/3 [00:00<?, ?it/s]

  0%|          | 0/17 [00:00<?, ?it/s]

  0%|          | 0/3 [00:00<?, ?it/s]

  0%|          | 0/17 [00:00<?, ?it/s]

  0%|          | 0/3 [00:00<?, ?it/s]

  0%|          | 0/17 [00:00<?, ?it/s]

  0%|          | 0/3 [00:00<?, ?it/s]

  0%|          | 0/17 [00:00<?, ?it/s]

  0%|          | 0/3 [00:00<?, ?it/s]
```

In [18]:
```
y_pred_db = model_db.predict(X_test_db)
```

In [19]:
```
sklearn.metrics.r2_score(y_test_db, y_pred_db)
```

Out[19]: 0.04194613859675578

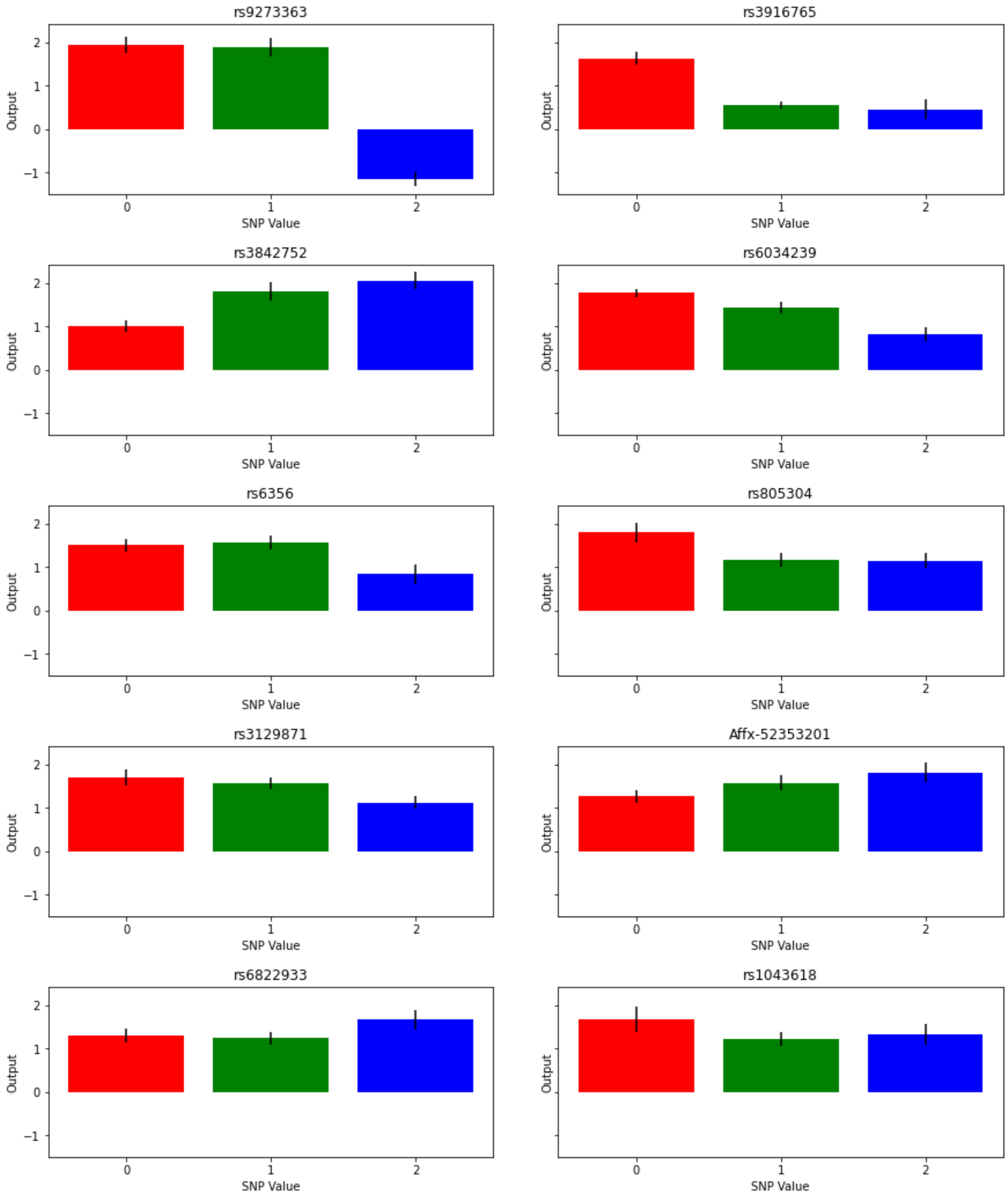
In [20]:
```
sklearn.metrics.mean_squared_error(y_test_db, y_pred_db)
```

Out[20]: 165.64681447018447

In [26]:
```
sorted_dict = get_importance(X_train_db, model_db)
```

In [28]:
```
snp_names = pd.read_csv('snp_names/diabetes_snp_names.csv')
indices = [str(i) for i in list(snp_names.index)]
names = list(snp_names['snp'])
snp_names_dict = dict(zip(indices, names))
```

In [29]:
```
barplot(sorted_dict, 'diabetes', model_db, snp_names_dict)
```



XGboost

In [ ]:
```
ebm = ExplainableBoostingClassifier(random_state=1, interactions=100)
print('fitting')
ebm.fit(X_train_red, y_train_red)
#print('explain_global')
#ebm_global = ebm.explain_global()
#show([ebm_local])
#print('explain_local')
#ebm_local = ebm.explain_local(X_test_red[:5], y_test_red[:5])
#show(ebm_local)
```

In [ ]:
```
preds = ebm.predict(X_test_red)
preds_list = [float(i) for i in preds]
y_test_red_list = [float(i) for i in list(y_test_red.values)]
fpr, tpr, threshold = metrics.roc_curve(y_test_red_list, preds_list)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

Regression bilirubin

In [ ]:
```
bilirubin = pd.read_csv('~/sasha_jess/cleaned_data/INI30840_data_pcs.csv')
```

In [ ]:
```
bilirubin
```

In [ ]:
```
X_train = bilirubin.iloc[:, 2:-1]
y_train = bilirubin.iloc[:, -1]
X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.2)
```

In [ ]:
```
X_train
```

In [ ]:
```
model_reg = NAMRegressor(
            num_epochs = 15,
            num_learners= 1,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            device = 'cuda',
            save_model_frequency = 5
)
model_reg.fit(X_train, y_train)
```

In [ ]:
```
y_pred = model_reg.predict(X_test)
```

In [ ]:
```
sklearn.metrics.r2_score(y_test, y_pred)
```

In [ ]:
```
sklearn.metrics.mean_squared_error(y_test, y_pred)
```

In [ ]:
```
df = pd.DataFrame(columns = ['mean', 'std', 'name'])
```

In [ ]:
```
#map_location=torch.device('cpu')
model=torch.load('output/0/ckpts/model-15.pt', map_location=torch.device('cpu'))
```

In [ ]:
```
print("Model's state_dict:")
for param_tensor in model['model_state_dict']:
    name = param_tensor
    mean = torch.mean(torch.abs(model['model_state_dict'][param_tensor]))
    std = torch.std(model['model_state_dict'][param_tensor])
    df.loc[len(df.index)] = [mean, std, name]
```

In [ ]:
```
df.sort_values("mean", ascending=False)[:10]
```

Classification Celiac Disease

In [10]:
```
celiac = pd.read_csv('~/sasha_jess/cleaned_data/HC303_data_pcs.csv')
celiac_nan = celiac.replace(-9, np.nan)
celiac = celiac_nan.fillna(celiac_nan.median())
```

In [11]:
```
celiac
```

Out[11]:

	Unnamed: 0	IID	gender	PC0	PC1	PC2	PC3	PC4	PC5	PC6	...	800609	800810	801676	801897	802098	8
0	0	1000370	1	-1.217298	-2.465534	9.275654	0.473448	-0.495340	-1.355526	2.292859	...	0.0	1.0	1.0	1.0	1.0	
1	1	1000998	1	5.831720	1.449414	0.872061	0.034517	3.357673	5.772951	-0.004874	...	1.0	1.0	1.0	0.0	1.0	
2	2	1001904	1	-8.586012	13.038083	-2.483975	3.420657	-3.249482	3.732272	-1.205457	...	0.0	1.0	1.0	1.0	1.0	
3	3	1001962	1	-0.694355	-1.732359	-1.275282	0.207636	-1.121330	0.782068	-0.299743	...	2.0	0.0	1.0	1.0	2.0	
4	4	1002535	1	1.810307	-0.264882	0.214023	0.326418	0.402879	-3.676624	10.926567	...	2.0	2.0	1.0	2.0	1.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
6383	6383	6019403	1	1.248860	0.037597	9.414793	-0.107599	-1.597313	-4.257469	-1.182120	...	0.0	2.0	2.0	1.0	1.0	
6384	6384	6021238	0	-2.291755	0.653849	-1.723715	4.957746	-5.634397	4.913494	-0.232687	...	0.0	0.0	0.0	0.0	2.0	
6385	6385	6021441	0	0.308161	-1.886748	-1.930490	-1.820300	1.437775	-1.208758	7.016675	...	2.0	0.0	2.0	2.0	0.0	
6386	6386	6022243	1	-1.599610	-0.878350	-1.405513	0.178063	-0.505243	0.460698	-0.154089	...	1.0	1.0	1.0	1.0	1.0	
6387	6387	6022298	1	-0.825470	-0.676547	-1.695263	0.540122	0.232044	-0.862867	1.204643	...	0.0	0.0	0.0	2.0	1.0	

6388 rows × 437 columns

In [12]:
```
X_train_celiac = celiac.iloc[:, 2:-1]
y_train_celiac = celiac.iloc[:, -1]
X_train_celiac, X_test_celiac, y_train_celiac, y_test_celiac = train_test_split(X_train_celiac, y_train_celiac, test_s
```

In [13]:
```
model_celiac = NAMClassifier(
            num_epochs=50,
            num_learners=10,
            early_stop_mode='max',
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_celiac.fit(X_train_celiac, y_train_celiac)
```

```
  0%|          | 0/50 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
0%|          | 0/1 [00:00<?, ?it/s]
0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
```

```
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/50 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
  0%|          | 0/1 [00:00<?, ?it/s]
  0%|          | 0/5 [00:00<?, ?it/s]
```

```
  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]

  0%|          | 0/5 [00:00<?, ?it/s]

  0%|          | 0/1 [00:00<?, ?it/s]
```

Out[13]: <nam.wrapper.wrapper.NAMClassifier at 0x7ef90e41bf50>

In [14]:
```
import sklearn.metrics as metrics
# calculate the fpr and tpr for all thresholds of the classification
preds = model_celiac.predict_proba(X_test_celiac)
fpr, tpr, threshold = metrics.roc_curve(y_test_celiac, preds)
roc_auc = metrics.auc(fpr, tpr)
print(roc_auc)
```

```
0.8517897160211116
```

In [15]:
```
sorted_dict = get_importance(X_train_celiac, model_celiac)
```

In [28]:
```
print(sorted_dict)
```


```
{'289538': [0.75464666, 162], '288739': [0.6613277, 155], '289519': [0.53644145, 160], '289528': [0.49612862, 161],
'287318': [0.44925427, 152], '288404': [0.36554137, 154], 'gender': [0.25559366, 0], '289841': [0.25318748, 165], '2
89589': [0.24827401, 164], '606603': [0.19352584, 320], '290508': [0.19176316, 170], '207950': [0.18848431, 116], '2
89587': [0.1833215, 163], '140664': [0.17885731, 78], '290307': [0.17344935, 167], '290509': [0.15195028, 171], '408
868': [0.14630117, 224], '199811': [0.1419222, 112], '315155': [0.14151067, 186], '478738': [0.1406427, 249], '19611
5': [0.13435864, 110], '290534': [0.13434368, 172], '781102': [0.13430819, 402], '696696': [0.13247594, 358], '28999
9': [0.12845299, 166], '285062': [0.12786497, 149], '659087': [0.11747029, 331], '381796': [0.11526509, 217], '77830
0': [0.11477302, 401], '132347': [0.11424939, 71], '315152': [0.108392216, 185], '284017': [0.10741344, 148], '18357
8': [0.10366282, 105], '498203': [0.09904869, 265], '845': [0.096910164, 11], '685694': [0.09556171, 352], '174182':
[0.09323145, 97], '427474': [0.09242032, 232], '82482': [0.09012121, 52], '312486': [0.08936283, 183], '73005': [0.0
8885216, 45], '170904': [0.088149786, 96], '266190': [0.088116586, 141], '467136': [0.08696777, 246], '43476': [0.08
610892, 35], '487726': [0.08585031, 257], '765261': [0.08543453, 390], '312475': [0.085097596, 182], '96730': [0.084
67747, 56], '34512': [0.08231247, 23], '255178': [0.081334576, 137], '19624': [0.07926964, 18], '610358': [0.0786224
6, 321], '803955': [0.0785678, 432], '636860': [0.07705086, 325], '286334': [0.076686546, 150], '766522': [0.0759417
64, 391], '787315': [0.07516797, 411], '207924': [0.07503868, 115], '359611': [0.074364744, 209], '785688': [0.07376
757, 407], '174183': [0.07313795, 98], '547356': [0.07242387, 292], '329276': [0.072411835, 198], '794508': [0.07215
883, 420], '791041': [0.07041262, 415], '359585': [0.06984344, 208], '51496': [0.06879893, 39], '590416': [0.0666340
4, 307], '497631': [0.06592429, 264], '259483': [0.06554212, 138], '290717': [0.06476963, 173], '756571': [0.0646246
8, 385], '555827': [0.06308909, 296], '3096': [0.06284897, 12], '287522': [0.0626409, 153], '46291': [0.062454417, 3
```


In [29]:
```
idxs = list(celiac.columns)[13:]
idxs
```

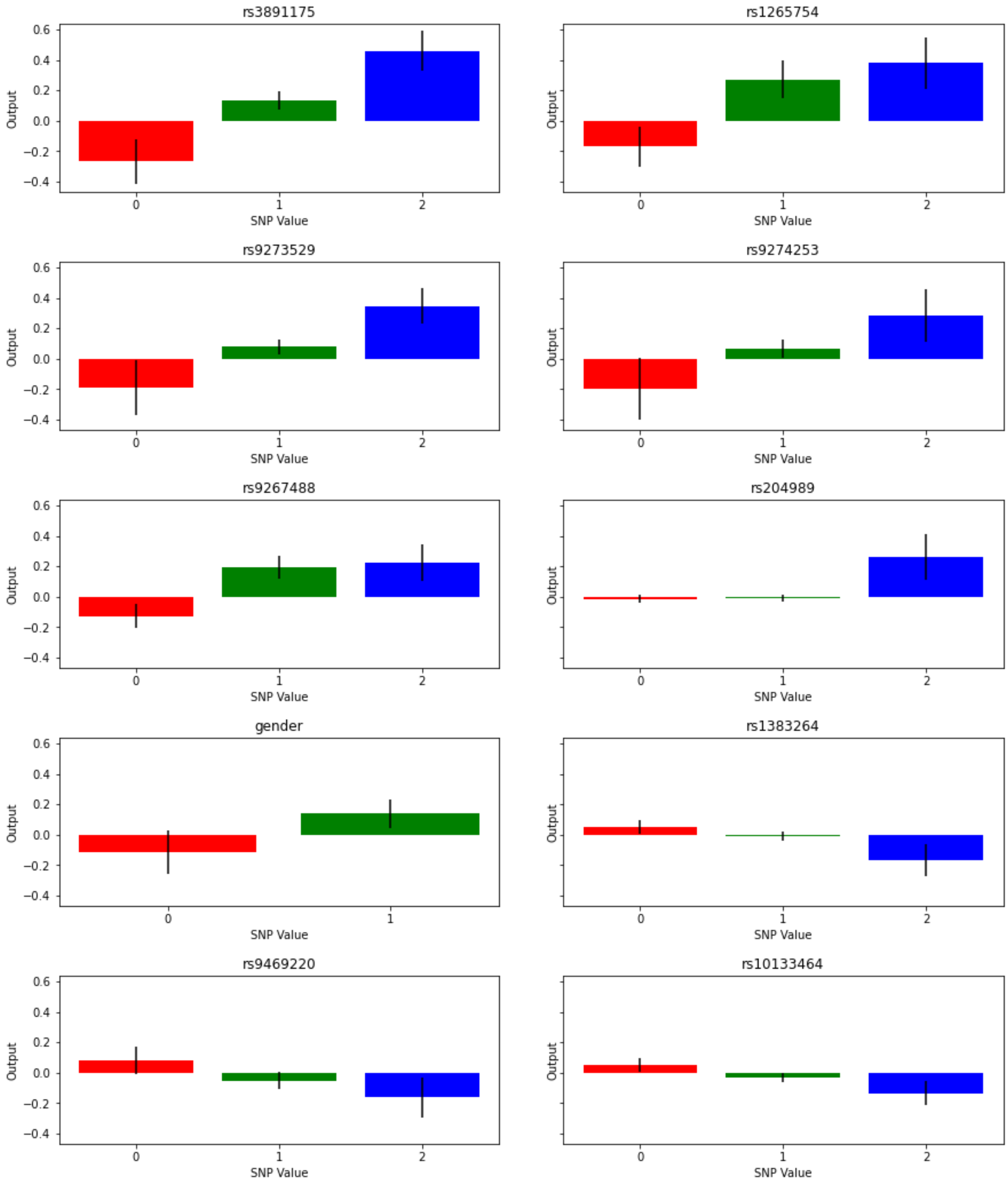
Out[29]:
```
['845',
 '3096',
 '3504',
 '6887',
 '9431',
 '12294',
 '16352',
 '19624',
 '20098',
 '21908',
 '30217',
 '30373',
 '34512',
 '35007',
 '35321',
 '36725',
 '37171',
 '37474',
 '37604',
```

In [30]:
```
snp_names = pd.read_csv('snp_names/celiac_snp_names.csv')
indices = [str(i) for i in list(snp_names.index)]
names = list(snp_names['snp'])
snp_names_dict = {idxs[i]:names[i] for i in range(len(indices))}
```

In [31]:
```
snp_names_dict
```

Out[31]:
```
{'845': 'rs3748816',
 '3096': 'rs12727642',
 '3504': 'rs12405873',
 '6887': 'rs694214',
 '9431': 'rs196432',
 '12294': 'rs1880418',
 '16352': 'Affx-35292281',
 '19624': 'rs6691768',
 '20098': 'rs11208062',
 '21908': 'rs560827',
 '30217': 'rs2336645',
 '30373': 'Affx-52321779',
 '34512': 'Affx-5303414',
 '35007': 'rs7545406',
 '35321': 'rs1055935',
 '36725': 'rs11264498',
 '37171': 'rs2778009',
 '37474': 'rs863362',
 '37604': 'rs857827',
```

In [34]:
```
barplot(sorted_dict, 'celiac', model_celiac,snp_names_dict)
```



In [ ]:
```

```

Multitask Classification

In [ ]:
```
merged_final = pd.read_csv('~/sasha_jess/cleaned_data/merged_data_pcs.csv').drop('Unnamed: 0', axis = 1)
X_train = merged_final.drop(['HC303',  'BIN_FC2001747'], axis = 1)
y_train = merged_final[['HC303', 'BIN_FC2001747']]
```

In [ ]:
```
X_train_merged, X_test_merged, y_train_merged, y_test_merged = \
train_test_split(X_train, y_train, test_size=0.2)
```

In [ ]:
```
model_merged = MultiTaskNAMClassifier(
            num_epochs=10,
            num_learners=1,
            early_stop_mode='max',
            num_subnets=1,
            monitor_loss=True,
            metric = 'auroc',
            n_jobs=1,
            device = 'cuda',
            save_model_frequency = 5
        )

model_merged.fit(X_train_merged, y_train_merged)
```

In [ ]:
```
pred = model_merged.predict_proba(X_test_merged)
```

In [ ]:
```
y_test_mtl = y_test_merged
y_test_mtl_flat = y_test_mtl.to_numpy().reshape(-1)
pred_flat = pred.reshape(-1)

non_nan_indices = y_test_mtl_flat == y_test_mtl_flat
y_test_mtl_flat = y_test_mtl_flat[non_nan_indices]
pred_flat = pred_flat[non_nan_indices]
```

In [ ]:
```
sk_metrics.roc_auc_score(y_test_mtl_flat, pred_flat)
```

In [ ]:
```
fpr, tpr, threshold = metrics.roc_curve(y_test_mtl_flat, pred_flat)
```

Lipoprotein A Regression

In [ ]:
```
lpa = pd.read_csv('~/sasha_jess/cleaned_data/INI30790_data_pcs.csv')
```

In [ ]:
```
lpa
```

In [ ]:
```
X_train_lpa
```

In [ ]:
```
X_train_lpa['8298']
```

In [ ]:
```
X_train_lpa = lpa.iloc[:, 2:-1]
y_train_lpa = lpa.iloc[:, -1]
X_train_lpa, X_test_lpa, y_train_lpa, y_test_lpa = train_test_split(X_train_lpa, y_train_lpa, test_size=0.2)
```

In [ ]:
```
model_lpa = NAMRegressor(
            num_epochs = 10,
            num_learners= 1,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            device = 'cuda',
            save_model_frequency = 5
)
model_lpa.fit(X_train_lpa, y_train_lpa)
```

In [ ]:
```
y_pred_lpa = model_lpa.predict(X_test_lpa)
```

In [ ]:
```
y_pred_lpa
```

In [ ]:
```
y_test_lpa
```

In [ ]:
```
sklearn.metrics.r2_score(y_test_lpa, y_pred_lpa)
```

In [ ]:
```
feature_predictions = get_feature_predictions(model_lpa, unique_features)
```

In [ ]:
```

```

Multitask Bilirubin and Diabetes Regression

In [6]:
```
merged = pd.read_csv('~/sasha_jess/cleaned_data/INI2976_INI30840_data_pcs.csv').drop('Unnamed: 0', axis = 1)
X_train = merged.drop(['INI2976',  'INI30840'], axis = 1)
y_train = merged[['INI2976', 'INI30840']]
```

In [7]:
```
X_train_merged, X_test_merged, y_train_merged, y_test_merged = \
train_test_split(X_train, y_train, test_size=0.2)
```

In [9]:
```
model_merged = MultiTaskNAMRegressor(
            num_epochs = 30,
            num_learners= 3,
            early_stop_mode='min',
            monitor_loss = True,
            metric = 'mse',
            n_jobs = 1,
            num_subnets=2,
            device = 'cuda',
            save_model_frequency = 5
        )

model_merged.fit(X_train_merged, y_train_merged)
```

```
  0%|          | 0/30 [00:00<?, ?it/s]
  0%|          | 0/17 [00:00<?, ?it/s]
  0%|          | 0/3 [00:00<?, ?it/s]
  0%|          | 0/17 [00:00<?, ?it/s]
  0%|          | 0/3 [00:00<?, ?it/s]
  0%|          | 0/17 [00:00<?, ?it/s]
  0%|          | 0/3 [00:00<?, ?it/s]
  0%|          | 0/17 [00:00<?, ?it/s]
  0%|          | 0/3 [00:00<?, ?it/s]
  0%|          | 0/17 [00:00<?, ?it/s]
  0%|          | 0/3 [00:00<?, ?it/s]
```

In [ ]:
```
y_pred_merged = model_merged.predict(X_test_merged)
```

In [ ]:
```
sklearn.metrics.r2_score(y_test_merged, y_pred_merged)
```

In [ ]:
```
sklearn.metrics.mean_squared_error(y_test_merged, y_pred_merged)
```

In [64]:
```
y_test_merged
```

Out[64]:

	INI2976	INI30840
7228	64.0	6.86
1510	45.0	6.73
19793	50.0	13.27
20996	62.0	11.41
2004	65.0	11.30
...	...	...
20544	45.0	5.50
22221	45.0	6.42
4516	50.0	8.04
2796	67.0	7.14
14592	50.0	13.34

4823 rows × 2 columns

In [75]:
```
from sklearn.metrics import mean_absolute_error
mean_absolute_error(y_test_merged, y_pred_merged)
```

Out[75]: 6.620096225742965

TOTAL RESULTS

In [ ]:
```
%load_ext tensorboard
```

In [ ]:

Model without PCA: AUC RED HAIR = 0.968018375610159 MSE BILIRUBIN = 10.557112893903545 AUROC CELIAC = 0.859

Model with PCA: AUC RED HAIR = 0.954682448362645 MSE BILIRUBIN = 14.351252695419824 AUC CELIAC = 0.856

In [ ]:

In [ ]: