In [1]:
```
import pandas as pd
from chembl_webresource_client.new_client import new_client
```

In [3]:
```
target = new_client.target
target_query = target.search('CHEMBL288')
targets = pd.DataFrame.from_dict(target_query)
targets
```

Out[3]:

	cross_references	organism	pref_name	score	species_group_flag	target_chembl_id	target_components	target_type	tax_id
0	[]	Homo sapiens	Phosphodiesterase 4D	11.0	False	CHEMBL288	[{'accession': 'Q08499', 'component_descriptio...	SINGLE PROTEIN	9606
1	[]	Homo sapiens	Phosphodiesterase 4	5.0	False	CHEMBL2093863	[{'accession': 'P27815', 'component_descriptio...	PROTEIN FAMILY	9606
2	[]	Homo sapiens	Phosphodiesterase; PDE3 & PDE4	4.0	False	CHEMBL2095153	[{'accession': 'P27815', 'component_descriptio...	SELECTIVITY GROUP	9606
3	[]	Homo sapiens	Phosphodiesterase 4 and 5 (PDE4 and PDE5)	4.0	False	CHEMBL2111340	[{'accession': 'O76074', 'component_descriptio...	SELECTIVITY GROUP	9606
4	[]	Homo sapiens	3',5'-cyclic phosphodiesterase	1.0	False	CHEMBL2363066	[{'accession': 'O76074', 'component_descriptio...	PROTEIN FAMILY	9606

In [6]:
```
selected_target = targets.target_chembl_id[0]
selected_target
```

Out[6]:
```
'CHEMBL288'
```

In [9]:
```
activity = new_client.activity
res = activity.filter(target_chembl_id=selected_target).filter(standard_type="IC50")
```

In [12]:
```
df = pd.DataFrame.from_dict(res)
```

In [15]:
```
print(len(res))
```

```
1708
```

In [40]:
```
df.head()
```

Out[40]:

	action_type	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_description	assay_type	assay_variant_accession	assay_
0	None	None	311417	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
1	None	None	315297	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
2	None	None	316479	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
3	None	None	904630	[]	CHEMBL761825	Evaluated for its ability to inhibit PDE4D.	B	None	
4	None	None	904635	[]	CHEMBL761825	Evaluated for its ability to inhibit PDE4D.	B	None	

5 rows × 46 columns

In [42]:
```
df.tail()
```

Out[42]:

	action_type	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_description	assay_type	assay_variant_accession	a
1703	{'action_type': 'INHIBITOR', 'description': 'N...	None	25627038	[]	CHEMBL5364574	Inhibition of human recombinant PDE4D3 express...	B	None	
1704	{'action_type': 'INHIBITOR', 'description': 'N...	None	25710748	[]	CHEMBL5388187	Inhibition of PDE4D (unknown origin)	B	None	
1705	{'action_type': 'INHIBITOR', 'description': 'N...	None	25710749	[]	CHEMBL5388187	Inhibition of PDE4D (unknown origin)	B	None	
1706	{'action_type': 'INHIBITOR', 'description': 'N...	None	25710752	[{'comments': None, 'relation': '=', 'result_f...	CHEMBL5388190	Inhibition of human full length PDE4D2 assesse...	B	None	
1707	None	None	25787617	[{'comments': None, 'relation': None, 'result_...	CHEMBL5474457	Selectivity interaction (Phosphodiesterase 4D,...	B	None	

5 rows × 46 columns

In [44]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1708 entries, 0 to 1707
Data columns (total 46 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   action_type                 334 non-null    object
 1   activity_comment            163 non-null    object
 2   activity_id                 1708 non-null   int64
 3   activity_properties         1708 non-null   object
 4   assay_chembl_id             1708 non-null   object
 5   assay_description           1708 non-null   object
 6   assay_type                  1708 non-null   object
 7   assay_variant_accession     0 non-null      object
 8   assay_variant_mutation      0 non-null      object
 9   bao_endpoint                1708 non-null   object
 10  bao_format                  1708 non-null   object
 11  bao_label                   1708 non-null   object
 12  canonical_smiles            1708 non-null   object
 13  data_validity_comment       30 non-null     object
 14  data_validity_description   30 non-null     object
 15  document_chembl_id          1708 non-null   object
 16  document_journal            1660 non-null   object
 17  document_year               1708 non-null   int64
 18  ligand_efficiency           1325 non-null   object
 19  molecule_chembl_id          1708 non-null   object
 20  molecule_pref_name          172 non-null    object
 21  parent_molecule_chembl_id   1708 non-null   object
 22  pchembl_value               1395 non-null   object
 23  potential_duplicate         1708 non-null   int64
 24  qudt_units                  1575 non-null   object
 25  record_id                   1708 non-null   int64
 26  relation                    1575 non-null   object
 27  src_id                      1708 non-null   int64
 28  standard_flag               1708 non-null   int64
 29  standard_relation           1575 non-null   object
 30  standard_text_value         0 non-null      object
 31  standard_type               1708 non-null   object
 32  standard_units              1575 non-null   object
 33  standard_upper_value        0 non-null      object
 34  standard_value              1575 non-null   object
 35  target_chembl_id            1708 non-null   object
 36  target_organism             1708 non-null   object
```

```
 37  target_pref_name           1708 non-null   object
 38  target_tax_id              1708 non-null   object
 39  text_value                 0 non-null      object
 40  toid                       0 non-null      object
 41  type                       1708 non-null   object
 42  units                      1508 non-null   object
 43  uo_units                   1575 non-null   object
 44  upper_value                3 non-null      object
 45  value                      1575 non-null   object
dtypes: int64(6), object(40)
memory usage: 613.9+ KB
```

In [47]:
```
df.columns
```

Out[47]:
```
Index(['action_type', 'activity_comment', 'activity_id', 'activity_properties',
       'assay_chembl_id', 'assay_description', 'assay_type',
       'assay_variant_accession', 'assay_variant_mutation', 'bao_endpoint',
       'bao_format', 'bao_label', 'canonical_smiles', 'data_validity_comment',
       'data_validity_description', 'document_chembl_id', 'document_journal',
       'document_year', 'ligand_efficiency', 'molecule_chembl_id',
       'molecule_pref_name', 'parent_molecule_chembl_id', 'pchembl_value',
       'potential_duplicate', 'qudt_units', 'record_id', 'relation', 'src_id',
       'standard_flag', 'standard_relation', 'standard_text_value',
       'standard_type', 'standard_units', 'standard_upper_value',
       'standard_value', 'target_chembl_id', 'target_organism',
       'target_pref_name', 'target_tax_id', 'text_value', 'toid', 'type',
       'units', 'uo_units', 'upper_value', 'value'],
      dtype='object')
```

In [50]:
```
df2 = df.dropna(subset=["standard_value", "canonical_smiles"])
```

In [52]:
```
df2
```

Out[52]:

	action_type	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_description	assay_type	assay_variant_accession	a
0	None	None	311417	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
1	None	None	315297	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
2	None	None	316479	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
3	None	None	904630	[]	CHEMBL761825	Evaluated for its ability to inhibit PDE4D.	B	None	
4	None	None	904635	[]	CHEMBL761825	Evaluated for its ability to inhibit PDE4D.	B	None	
...	...	...	...	...	...	...	...	...	
1703	{'action_type': 'INHIBITOR', 'description': 'N...	None	25627038	[]	CHEMBL5364574	Inhibition of human recombinant PDE4D3 express...	B	None	
1704	{'action_type': 'INHIBITOR', 'description': 'N...	None	25710748	[]	CHEMBL5388187	Inhibition of PDE4D (unknown origin)	B	None	
1705	{'action_type': 'INHIBITOR', 'description': 'N...	None	25710749	[]	CHEMBL5388187	Inhibition of PDE4D (unknown origin)	B	None	

	action_type	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_description	assay_type	assay_variant_accession	a
1706	{'action_type': 'INHIBITOR', 'description': 'N...	None	25710752	[{'comments': None, 'relation': '=', 'result_f...	CHEMBL5388190	Inhibition of human full length PDE4D2 assesse...	B	None	
1707	None	None	25787617	[{'comments': None, 'relation': None, 'result_...	CHEMBL5474457	Selectivity interaction (Phosphodiesterase 4D,...	B	None	

1575 rows × 46 columns

In [56]:
```
len(df2.canonical_smiles.unique())
```

Out[56]: 1287

In [60]:
```
df2_nr = df2.drop_duplicates(subset="canonical_smiles", keep="first").reset_index(drop=True)
```

In [63]:
```
df2_nr
```

Out[63]:

	action_type	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_description	assay_type	assay_variant_accession	a
0	None	None	311417	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
1	None	None	315297	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
2	None	None	316479	[]	CHEMBL761828	Inhibition of human Phosphodiesterase 4D from ...	B	None	
3	None	None	904630	[]	CHEMBL761825	Evaluated for its ability to inhibit PDE4D.	B	None	
4	None	None	904635	[]	CHEMBL761825	Evaluated for its ability to inhibit PDE4D.	B	None	
...	...	...	...	...	...	...	...	...	...
1282	{'action_type': 'INHIBITOR', 'description': 'N...	None	25527403	[{'comments': None, 'relation': '=', 'result_f...	CHEMBL5339428	Inhibition of recombinant PDE4D2 (86 to 413 re...	B	None	
1283	{'action_type': 'INHIBITOR', 'description': 'N...	None	25527404	[{'comments': None, 'relation': '=', 'result_f...	CHEMBL5339428	Inhibition of recombinant PDE4D2 (86 to 413 re...	B	None	
1284	{'action_type': 'INHIBITOR', 'description': 'N...	None	25527405	[{'comments': None, 'relation': '=', 'result_f...	CHEMBL5339428	Inhibition of recombinant PDE4D2 (86 to 413 re...	B	None	

	action_type	activity_comment	activity_id	activity_properties	assay_chembl_id	assay_description	assay_type	assay_variant_accession	a
1285	{'action_type': 'INHIBITOR', 'description': 'N...	None	25527406	[{'comments': None, 'relation': '=', 'result_f...	CHEMBL5339428	Inhibition of recombinant PDE4D2 (86 to 413 re...	B	None	
1286	{'action_type': 'INHIBITOR', 'description': 'N...	None	25627038	[]	CHEMBL5364574	Inhibition of human recombinant PDE4D3 express...	B	None	

1287 rows × 46 columns

In [67]:
```
df2_nr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1287 entries, 0 to 1286
Data columns (total 46 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   action_type                 232 non-null    object
 1   activity_comment            17 non-null     object
 2   activity_id                 1287 non-null   int64
 3   activity_properties         1287 non-null   object
 4   assay_chembl_id             1287 non-null   object
 5   assay_description           1287 non-null   object
 6   assay_type                  1287 non-null   object
 7   assay_variant_accession     0 non-null      object
 8   assay_variant_mutation      0 non-null      object
 9   bao_endpoint                1287 non-null   object
 10  bao_format                  1287 non-null   object
 11  bao_label                   1287 non-null   object
 12  canonical_smiles            1287 non-null   object
 13  data_validity_comment       25 non-null     object
 14  data_validity_description   25 non-null     object
 15  document_chembl_id          1287 non-null   object
 16  document_journal            1257 non-null   object
 17  document_year               1287 non-null   int64
 18  ligand_efficiency           1065 non-null   object
 19  molecule_chembl_id          1287 non-null   object
 20  molecule_pref_name          51 non-null     object
 21  parent_molecule_chembl_id   1287 non-null   object
 22  pchembl_value               1118 non-null   object
 23  potential_duplicate         1287 non-null   int64
 24  qudt_units                  1287 non-null   object
 25  record_id                   1287 non-null   int64
 26  relation                    1287 non-null   object
 27  src_id                      1287 non-null   int64
 28  standard_flag               1287 non-null   int64
 29  standard_relation           1287 non-null   object
 30  standard_text_value         0 non-null      object
 31  standard_type               1287 non-null   object
 32  standard_units              1287 non-null   object
 33  standard_upper_value        0 non-null      object
 34  standard_value              1287 non-null   object
 35  target_chembl_id            1287 non-null   object
 36  target_organism             1287 non-null   object
```

```
 37  target_pref_name           1287 non-null   object
 38  target_tax_id              1287 non-null   object
 39  text_value                 0 non-null      object
 40  toid                       0 non-null      object
 41  type                       1287 non-null   object
 42  units                      1260 non-null   object
 43  uo_units                   1287 non-null   object
 44  upper_value                2 non-null      object
 45  value                      1287 non-null   object
dtypes: int64(6), object(40)
memory usage: 462.6+ KB
```

In [71]:
```
df2_nr .columns
```

Out[71]:
```
Index(['action_type', 'activity_comment', 'activity_id', 'activity_properties',
       'assay_chembl_id', 'assay_description', 'assay_type',
       'assay_variant_accession', 'assay_variant_mutation', 'bao_endpoint',
       'bao_format', 'bao_label', 'canonical_smiles', 'data_validity_comment',
       'data_validity_description', 'document_chembl_id', 'document_journal',
       'document_year', 'ligand_efficiency', 'molecule_chembl_id',
       'molecule_pref_name', 'parent_molecule_chembl_id', 'pchembl_value',
       'potential_duplicate', 'qudt_units', 'record_id', 'relation', 'src_id',
       'standard_flag', 'standard_relation', 'standard_text_value',
       'standard_type', 'standard_units', 'standard_upper_value',
       'standard_value', 'target_chembl_id', 'target_organism',
       'target_pref_name', 'target_tax_id', 'text_value', 'toid', 'type',
       'units', 'uo_units', 'upper_value', 'value'],
      dtype='object')
```

Data pre-processing of the bioactivity data

Combine the 3 columns (molecule_chembl_id,canonical_smiles,standard_value) and bioactivity_class into

In [76]:
```
selection = ['molecule_chembl_id','canonical_smiles','standard_value']
df3 = df2_nr[selection]
df3
```

Out[76]:

	molecule_chembl_id	canonical_smiles	standard_value
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	63.0
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	1.0
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	1.5
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	22.0
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	100000.0
...	...	...	...
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	20.2
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	9.3
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	6.5
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	8.2
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	0.3981

1287 rows × 3 columns

Labeling compounds as either being active, inactive or intermediate

The bioactivity data is in the IC50 unit. Compounds having values of less than 1000 nM will be considered to be active while those greater than 10,000 nM will be considered to be inactive. As for those values in between 1,000 and 10,000 nM will be referred to as intermediate.

In [81]:
```
bioactivity_threshold = []
for i in df3.standard_value:
  if float(i) >= 10000:
    bioactivity_threshold.append("inactive")
  elif float(i) <= 1000:
    bioactivity_threshold.append("active")
  else:
    bioactivity_threshold.append("intermediate")
```

In [84]:
```
bioactivity_class = pd.Series(bioactivity_threshold, name='class')
df5 = pd.concat([df3, bioactivity_class], axis=1)
df5
```

Out[84]:

	molecule_chembl_id	canonical_smiles	standard_value	class
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	63.0	active
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	1.0	active
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	1.5	active
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	22.0	active
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	100000.0	inactive
...	...	...	...	...
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	20.2	active
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	9.3	active
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	6.5	active
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	8.2	active
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	0.3981	active

1287 rows × 4 columns

In [93]:
```
df5.to_csv('Phosphodiesterase 4D_03_bioactivity_data_curated.csv', index=False)
```

In [96]:
```
df = pd.read_csv(r"C:\Users\manoj\OneDrive\Desktop\Phosphodiesterase 4D\Phosphodiesterase 4D_03_bioactivity_data_curated.csv")
df
```

Out[96]:

	molecule_chembl_id	canonical_smiles	standard_value	class
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	63.0000	active
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	1.0000	active
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	1.5000	active
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	22.0000	active
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	100000.0000	inactive
...	...	...	...	...
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	20.2000	active
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	9.3000	active
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	6.5000	active
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	8.2000	active
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	0.3981	active

1287 rows × 4 columns

In [100…
```
df_no_smiles = df.drop(columns='canonical_smiles')
```

In [103…
```
smiles = []

for i in df.canonical_smiles.tolist():
  cpd = str(i).split('.')
  cpd_longest = max(cpd, key = len)
  smiles.append(cpd_longest)

smiles = pd.Series(smiles, name = 'canonical_smiles')
```

In [106…
```
df_clean_smiles = pd.concat([df_no_smiles,smiles], axis=1)
df_clean_smiles
```

Out[106...

	molecule_chembl_id	standard_value	class	canonical_smiles
0	CHEMBL511115	63.0000	active	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1
1	CHEMBL74078	1.0000	active	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...
2	CHEMBL77826	1.5000	active	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1
3	CHEMBL97817	22.0000	active	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...
4	CHEMBL319809	100000.0000	inactive	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...
...	...	...	...	...
1282	CHEMBL5405731	20.2000	active	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...
1283	CHEMBL5435764	9.3000	active	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...
1284	CHEMBL5421333	6.5000	active	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...
1285	CHEMBL5440771	8.2000	active	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...
1286	CHEMBL5417689	0.3981	active	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...

1287 rows × 4 columns

Calculate Lipinski descriptors

Christopher Lipinski, a scientist at Pfizer, came up with a set of rule-of-thumb for evaluating the druglikeness of compounds. Such druglikeness is based on the Absorption, Distribution, Metabolism and Excretion (ADME) that is also known as the pharmacokinetic profile. Lipinski analyzed all orally active FDA-approved drugs in the formulation of what is to be known as the Rule-of-Five or Lipinski's Rule.

The Lipinski's Rule stated the following:

Molecular weight < 500 Dalton Octanol-water partition coefficient (LogP) < 5 Hydrogen bond donors < 5 Hydrogen bond acceptors < 10

In [ ]:

In [112...

```
import numpy as np
from rdkit import Chem
```

```
from rdkit.Chem import Descriptors, Lipinski
```

Calculate descriptors

In [118…

```
# Inspired by: https://codeocean.com/explore/capsules?query=tag:data-curation

def lipinski(smiles, verbose=False):

    moldata= []
    for elem in smiles:
        mol=Chem.MolFromSmiles(elem)
        moldata.append(mol)

    baseData= np.arange(1,1)
    i=0
    for mol in moldata:

        desc_MolWt = Descriptors.MolWt(mol)
        desc_MolLogP = Descriptors.MolLogP(mol)
        desc_NumHDonors = Lipinski.NumHDonors(mol)
        desc_NumHAcceptors = Lipinski.NumHAcceptors(mol)

        row = np.array([desc_MolWt,
                        desc_MolLogP,
                        desc_NumHDonors,
                        desc_NumHAcceptors])

        if(i==0):
            baseData=row
        else:
            baseData=np.vstack([baseData, row])
        i=i+1

    columnNames=["MW","LogP","NumHDonors","NumHAcceptors"]
    descriptors = pd.DataFrame(data=baseData,columns=columnNames)

    return descriptors
```

In [121…
```
df_lipinski = lipinski(df_clean_smiles.canonical_smiles)
df_lipinski
```

Out[121…

	MW	LogP	NumHDonors	NumHAcceptors
0	343.423	4.05278	1.0	4.0
1	371.352	4.57020	1.0	5.0
2	368.352	4.19820	1.0	6.0
3	464.562	4.81650	1.0	7.0
4	494.588	4.17900	2.0	8.0
...	...	...	...	...
1282	607.678	8.33980	0.0	7.0
1283	607.678	8.33980	0.0	7.0
1284	590.676	7.59570	0.0	8.0
1285	607.678	8.33980	0.0	7.0
1286	770.732	5.85120	2.0	11.0

1287 rows × 4 columns

Combine DataFrames

In [127…
```
df_lipinski
```

Out[127…

	MW	LogP	NumHDonors	NumHAcceptors
0	343.423	4.05278	1.0	4.0
1	371.352	4.57020	1.0	5.0
2	368.352	4.19820	1.0	6.0
3	464.562	4.81650	1.0	7.0
4	494.588	4.17900	2.0	8.0
...	...	...	...	...
1282	607.678	8.33980	0.0	7.0
1283	607.678	8.33980	0.0	7.0
1284	590.676	7.59570	0.0	8.0
1285	607.678	8.33980	0.0	7.0
1286	770.732	5.85120	2.0	11.0

1287 rows × 4 columns

In [129…

```
df_lipinski.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1287 entries, 0 to 1286
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   MW              1287 non-null   float64
 1   LogP            1287 non-null   float64
 2   NumHDonors      1287 non-null   float64
 3   NumHAcceptors   1287 non-null   float64
dtypes: float64(4)
memory usage: 40.3 KB
```

In [131…

```
df_lipinski.describe()
```

Out[131…

	MW	LogP	NumHDonors	NumHAcceptors
count	1287.000000	1287.000000	1287.000000	1287.000000
mean	409.690984	4.243414	1.129759	5.567988
std	89.449040	1.439778	1.120060	1.757411
min	164.595000	-0.074800	0.000000	1.000000
25%	345.606500	3.235800	0.000000	4.000000
50%	400.508000	4.097500	1.000000	5.000000
75%	463.504000	5.010950	2.000000	7.000000
max	900.903000	12.544800	9.000000	14.000000

In [134…

```
df
```

Out[134...

	molecule_chembl_id	canonical_smiles	standard_value	class
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	63.0000	active
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	1.0000	active
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	1.5000	active
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	22.0000	active
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	100000.0000	inactive
...	...	...	...	...
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	20.2000	active
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	9.3000	active
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	6.5000	active
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	8.2000	active
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	0.3981	active

1287 rows × 4 columns

In [136...

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1287 entries, 0 to 1286
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   molecule_chembl_id  1287 non-null   object 
 1   canonical_smiles    1287 non-null   object 
 2   standard_value      1287 non-null   float64
 3   class               1287 non-null   object 
dtypes: float64(1), object(3)
memory usage: 40.3+ KB
```

In [138...

```
df.describe()
```

Out[138...

	standard_value
count	1287.000000
mean	11005.779253
std	29937.136744
min	0.019950
25%	45.000000
50%	1000.000000
75%	10000.000000
max	602800.000000

combine the 2 DataFrame

In [143...

```
df_combined = pd.concat([df,df_lipinski], axis=1)
```

In [147...

```
df_combined
```

Out[147...

	molecule_chembl_id	canonical_smiles	standard_value	class	MW	LogP	NumHDonors	NumHAcc
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	63.0000	active	343.423	4.05278	1.0	
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	1.0000	active	371.352	4.57020	1.0	
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	1.5000	active	368.352	4.19820	1.0	
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	22.0000	active	464.562	4.81650	1.0	
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	100000.0000	inactive	494.588	4.17900	2.0	
...	...	...	...	...	...	...	...	
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	20.2000	active	607.678	8.33980	0.0	
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	9.3000	active	607.678	8.33980	0.0	
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	6.5000	active	590.676	7.59570	0.0	
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	8.2000	active	607.678	8.33980	0.0	
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	0.3981	active	770.732	5.85120	2.0	

1287 rows × 8 columns

1287 rows × 8 columns

In [149...

```
df_combined.head()
```

Out[149...

	molecule_chembl_id	canonical_smiles	standard_value	class	MW	LogP	NumHDonors	NumHAcceptors
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	63.0	active	343.423	4.05278	1.0	4.0
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	1.0	active	371.352	4.57020	1.0	5.0
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	1.5	active	368.352	4.19820	1.0	6.0
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	22.0	active	464.562	4.81650	1.0	7.0
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	100000.0	inactive	494.588	4.17900	2.0	8.0

In [151...

```
df_combined.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1287 entries, 0 to 1286
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   molecule_chembl_id  1287 non-null   object 
 1   canonical_smiles    1287 non-null   object 
 2   standard_value      1287 non-null   float64
 3   class               1287 non-null   object 
 4   MW                  1287 non-null   float64
 5   LogP                1287 non-null   float64
 6   NumHDonors          1287 non-null   float64
 7   NumHAcceptors       1287 non-null   float64
dtypes: float64(5), object(3)
memory usage: 80.6+ KB
```

In [153...

```
df_combined.describe()
```

Out[153...

	standard_value	MW	LogP	NumHDonors	NumHAcceptors
count	1287.000000	1287.000000	1287.000000	1287.000000	1287.000000
mean	11005.779253	409.690984	4.243414	1.129759	5.567988
std	29937.136744	89.449040	1.439778	1.120060	1.757411
min	0.019950	164.595000	-0.074800	0.000000	1.000000
25%	45.000000	345.606500	3.235800	0.000000	4.000000
50%	1000.000000	400.508000	4.097500	1.000000	5.000000
75%	10000.000000	463.504000	5.010950	2.000000	7.000000
max	602800.000000	900.903000	12.544800	9.000000	14.000000

Convert IC50 to pIC50

To allow IC50 data to be more uniformly distributed, we will convert IC50 to the negative logarithmic scale which is essentially -log10(IC50).

This custom function pIC50() will accept a DataFrame as input and will:

Take the IC50 values from the standard_value column and converts it from nM to M by multiplying the value by 10 Take the molar value and apply -log10 Delete the standard_value column and create a new pIC50 column

In [161...

```
# https://github.com/chaninlab/estrogen-receptor-alpha-qsar/blob/master/02_ER_alpha_RO5.ipynb

import numpy as np

def pIC50(input):
    pIC50 = []

    for i in input['standard_value_norm']:
        molar = i*(10**-9) # Converts nM to M
        pIC50.append(-np.log10(molar))

    input['pIC50'] = pIC50
    x = input.drop('standard_value_norm', 1)
```

```
    return x
```

In [164…
```
df_combined.standard_value.describe()
```

Out[164…
```
count      1287.000000
mean      11005.779253
std       29937.136744
min           0.019950
25%          45.000000
50%        1000.000000
75%       10000.000000
max      602800.000000
Name: standard_value, dtype: float64
```

In [167…
```
-np.log10( (10**-9)* 100000000 )
```

Out[167…
```
1.0
```

In [169…
```
-np.log10( (10**-9)* 10000000000 )
```

Out[169…
```
-1.0
```

In [180…
```
def norm_value(input):
    norm = []

    for i in input['standard_value']:
        if i > 100000000:
          i = 100000000
        norm.append(i)

    input['standard_value_norm'] = norm
    x = input.drop('standard_value', axis=1)

    return x
```

We will first apply the norm_value() function so that the values in the standard_value column is normalized.

In [183…]
```
df_norm = norm_value(df_combined)
df_norm
```

Out[183…]

	molecule_chembl_id	canonical_smiles	class	MW	LogP	NumHDonors	NumHAcceptors	standard
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	active	343.423	4.05278	1.0	4.0	
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	active	371.352	4.57020	1.0	5.0	
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	active	368.352	4.19820	1.0	6.0	
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	active	464.562	4.81650	1.0	7.0	
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	inactive	494.588	4.17900	2.0	8.0	
...	...	...	...	...	...	...	...	...
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0	
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0	
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	active	590.676	7.59570	0.0	8.0	
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	active	607.678	8.33980	0.0	7.0	
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	active	770.732	5.85120	2.0	11.0	

1287 rows × 8 columns

In [186…]
```
df_norm.standard_value_norm
```

Out[186…

```
0           63.0000
1            1.0000
2            1.5000
3           22.0000
4       100000.0000
           ...
1282        20.2000
1283         9.3000
1284         6.5000
1285         8.2000
1286         0.3981
Name: standard_value_norm, Length: 1287, dtype: float64
```

In [188…

```
df_norm.standard_value_norm.describe()
```

Out[188…

```
count      1287.000000
mean      11005.779253
std       29937.136744
min           0.019950
25%          45.000000
50%        1000.000000
75%       10000.000000
max      602800.000000
Name: standard_value_norm, dtype: float64
```

In [201…

```
df_final = df_norm.drop('standard_value_norm', axis=1)
df_final
```

Out[201...

	molecule_chembl_id	canonical_smiles	class	MW	LogP	NumHDonors	NumHAcceptors	pIC50
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	active	343.423	4.05278	1.0	4.0	7.200659
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	active	371.352	4.57020	1.0	5.0	9.000000
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	active	368.352	4.19820	1.0	6.0	8.823909
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	active	464.562	4.81650	1.0	7.0	7.657577
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	inactive	494.588	4.17900	2.0	8.0	4.000000
...	...	...	...	...	...	...	...	...
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0	7.694649
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0	8.031517
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	active	590.676	7.59570	0.0	8.0	8.187087
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	active	607.678	8.33980	0.0	7.0	8.086186
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	active	770.732	5.85120	2.0	11.0	9.400008

1287 rows × 8 columns

In [206...

```
df_final.pIC50.describe()
```

Out[206...

```
count    1287.000000
mean        6.233700
std         1.444116
min         3.219827
25%         5.000000
50%         6.000000
75%         7.346787
max        10.700057
Name: pIC50, dtype: float64
```

Removing the 'intermediate' bioactivity class

Here, we will be removing the intermediate class from our data set.

In [212…
```
df_2class = df_final[df_final["class"]  != 'intermediate']
df_2class
```

Out[212…

	molecule_chembl_id	canonical_smiles	class	MW	LogP	NumHDonors	NumHAcceptors	pIC50
0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	active	343.423	4.05278	1.0	4.0	7.200659
1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	active	371.352	4.57020	1.0	5.0	9.000000
2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	active	368.352	4.19820	1.0	6.0	8.823909
3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	active	464.562	4.81650	1.0	7.0	7.657577
4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	inactive	494.588	4.17900	2.0	8.0	4.000000
...	...	...	...	...	...	...	...	...
1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0	7.694649
1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0	8.031517
1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	active	590.676	7.59570	0.0	8.0	8.187087
1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	active	607.678	8.33980	0.0	7.0	8.086186
1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	active	770.732	5.85120	2.0	11.0	9.400008

983 rows × 8 columns

In [341…
```
df_2class.to_csv('Phosphodiesterase 4D_04_bioactivity_data_3class_pIC50.csv')
```

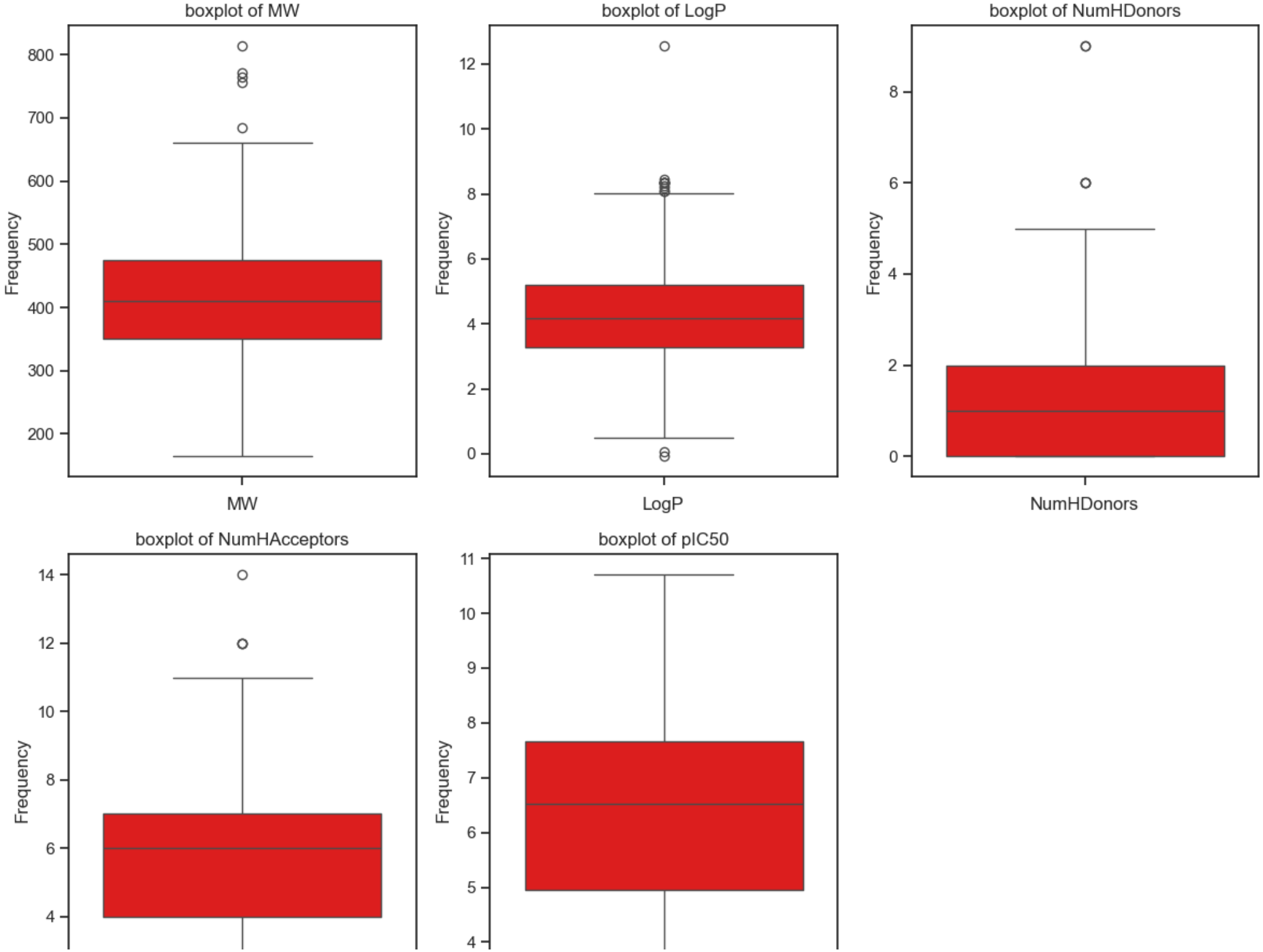
Exploratory Data Analysis (Chemical Space Analysis) via Lipinski descriptors

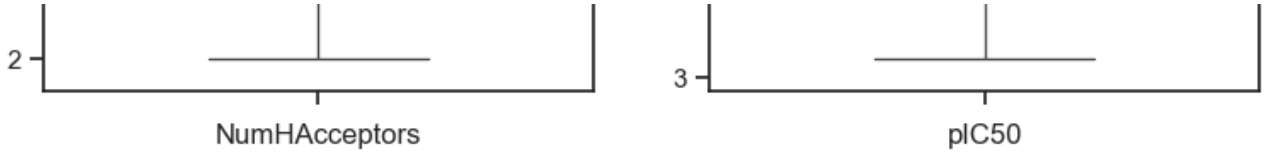
In [224…

```
import seaborn as sns
sns.set(style='ticks')
import matplotlib.pyplot as plt
```

In [226…

```
plt.figure(figsize=(12, 10))
numeric_columns = ['MW', 'LogP', 'NumHDonors', 'NumHAcceptors', 'pIC50']
for i, column in enumerate(numeric_columns, 1):
 plt.subplot(2, 3, i) # 2 rows and 3 columns for better layout
 sns.boxplot(df_2class[column],color='red') # kde=True for kernel density estimate
 plt.title(f'boxplot of {column}')
 plt.xlabel(column)
 plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```

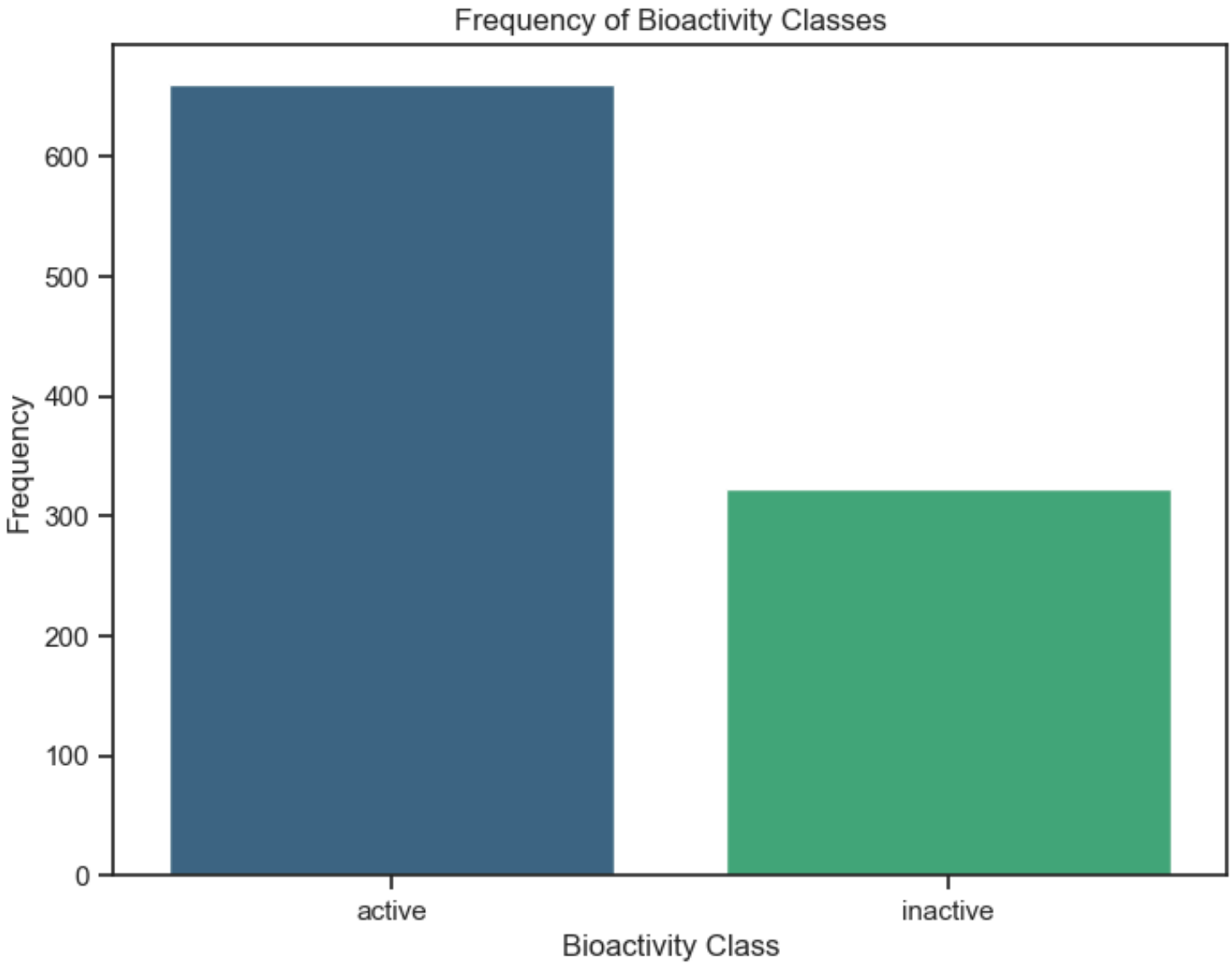

boxplot of MW
Frequency
MW
boxplot of LogP
Frequency
LogP
boxplot of NumHDonors
Frequency
NumHDonors
boxplot of NumHAcceptors
Frequency
boxplot of pIC50
Frequency



Frequency of Bioactivity Classes

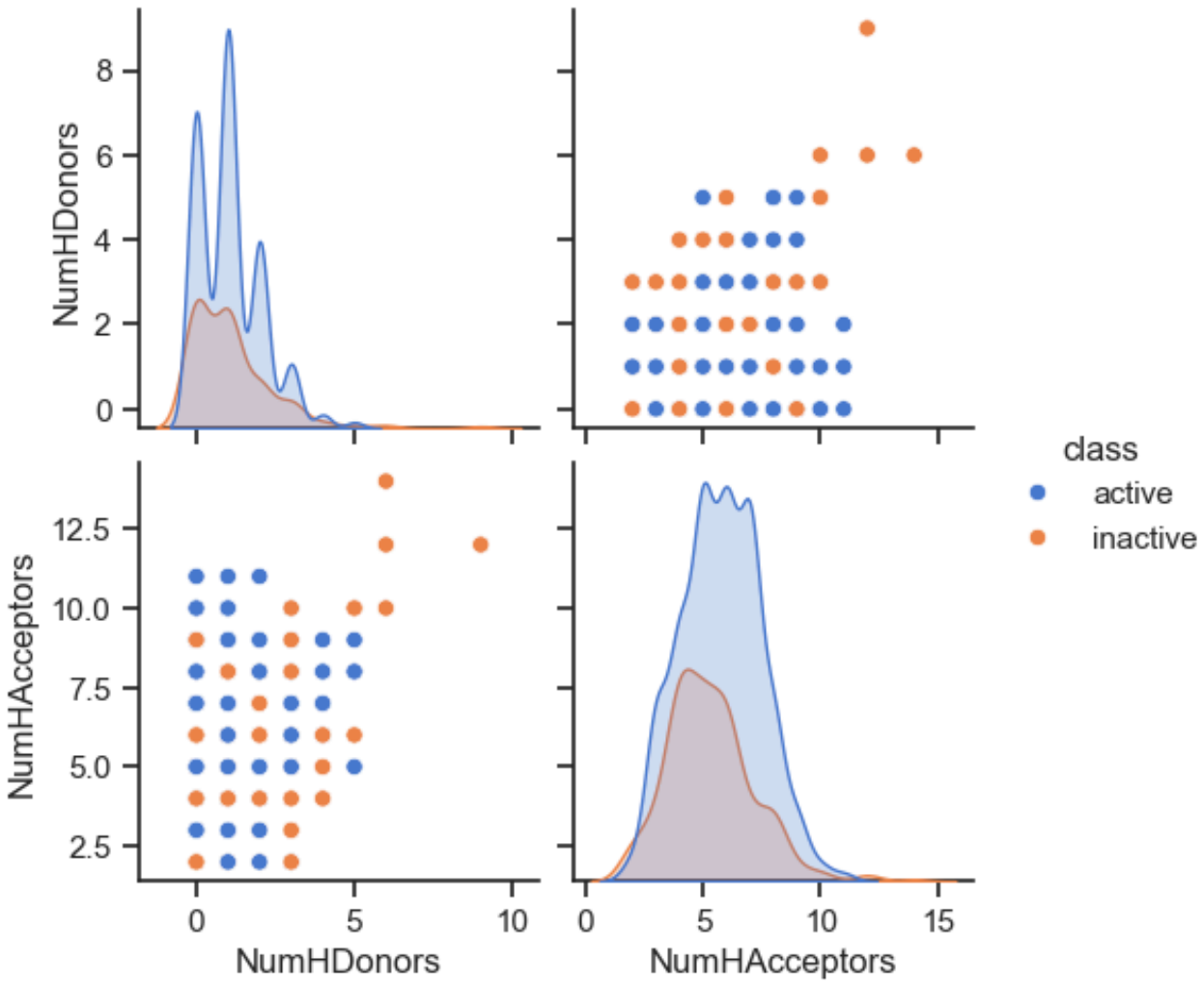
In [231…

```
# Set figure size
plt.figure(figsize=(8, 6))
# Count plot with hue
sns.countplot(x='class', hue='class', data=df_2class, palette='viridis')
# Add labels and title
plt.xlabel('Bioactivity Class')
plt.ylabel('Frequency')
plt.title('Frequency of Bioactivity Classes')
# Show plot
plt.show()
```



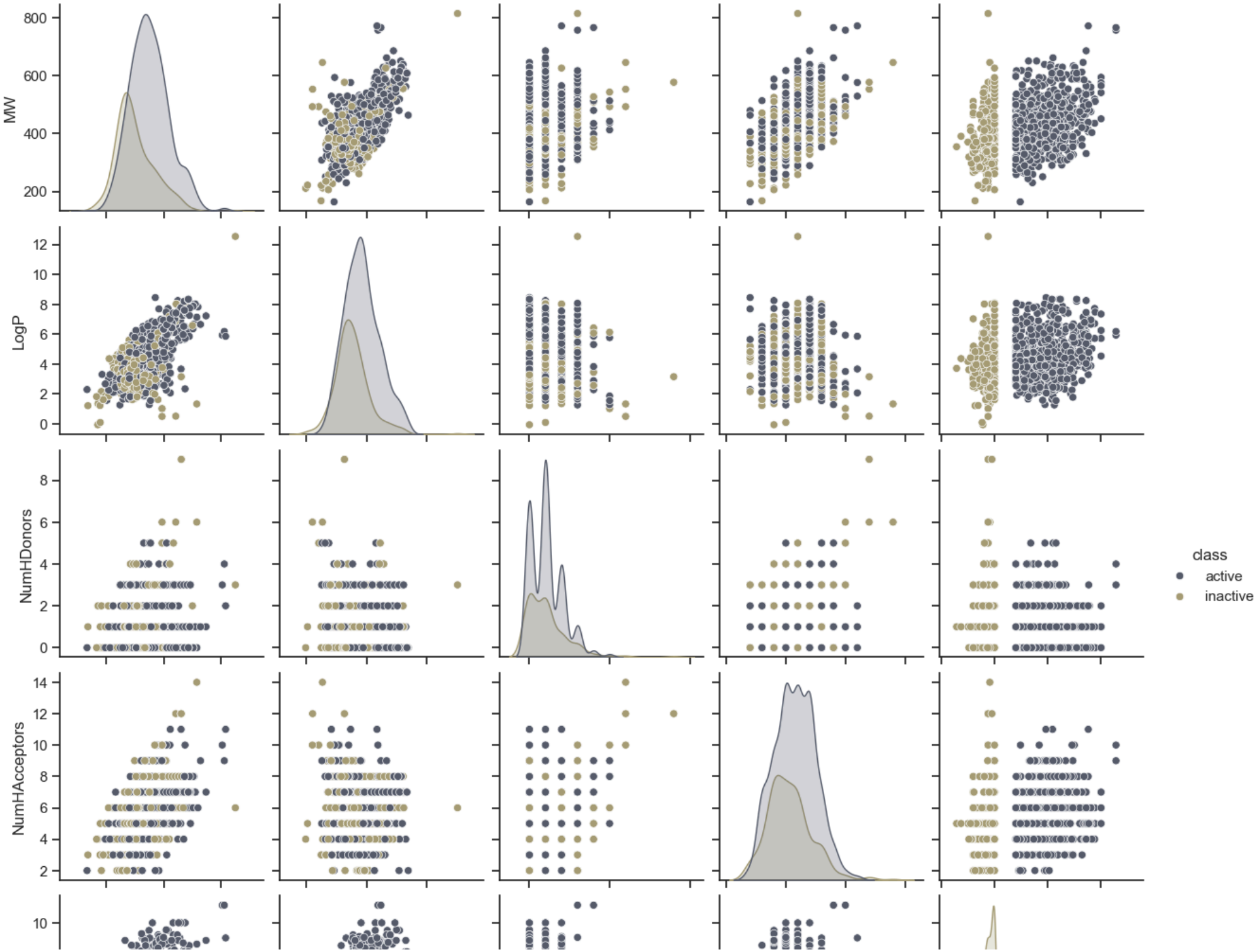
In [242…

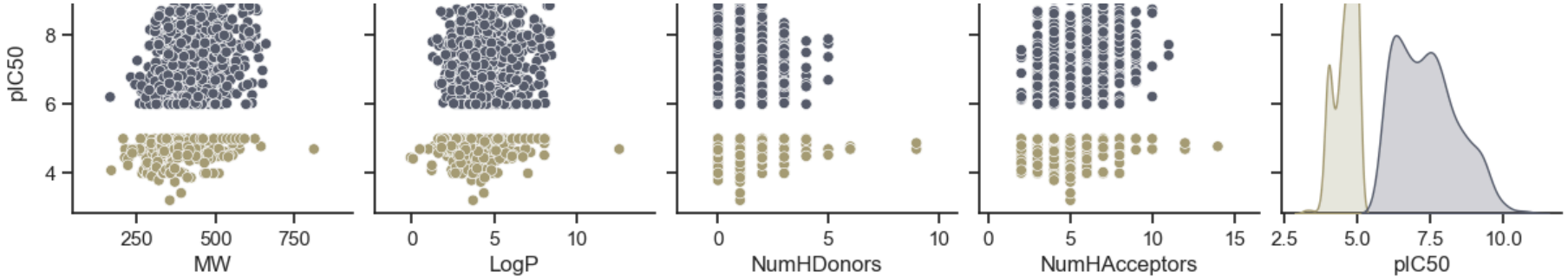
```
sns.pairplot(df_2class, vars=['NumHDonors', 'NumHAcceptors'], hue='class', palette='muted')
# Show plot
plt.show()
```



In [247…

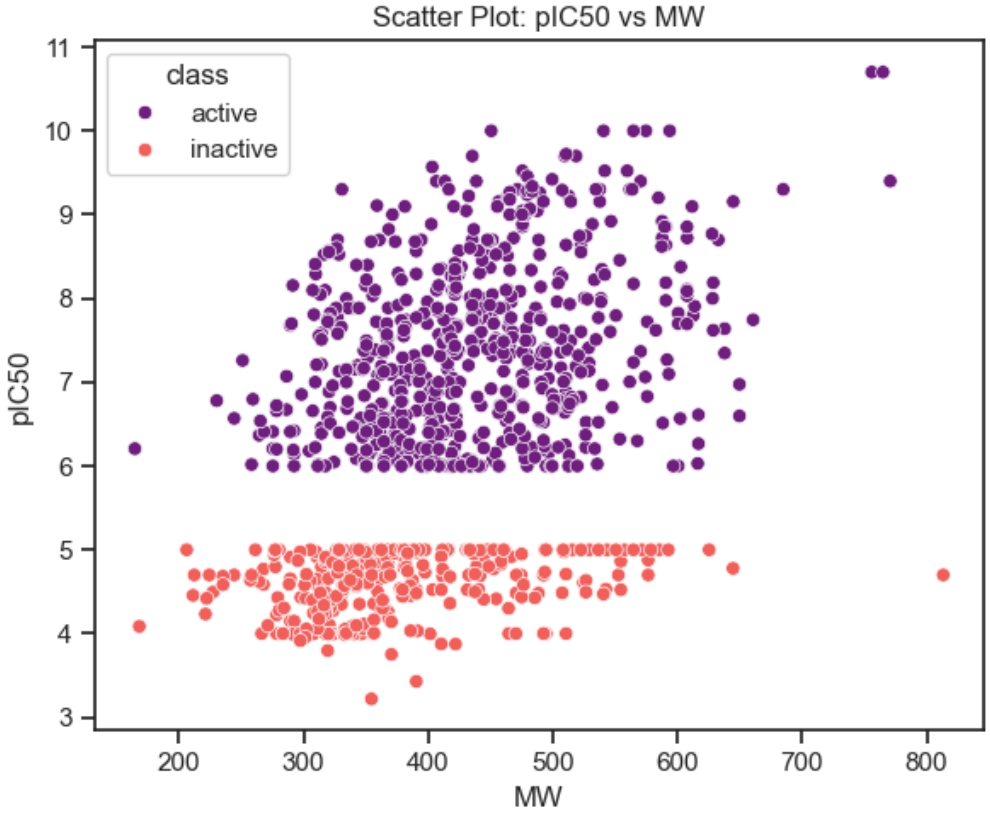
```
sns.pairplot(df_2class, hue='class', palette='cividis')
# Show plot
plt.show()
```

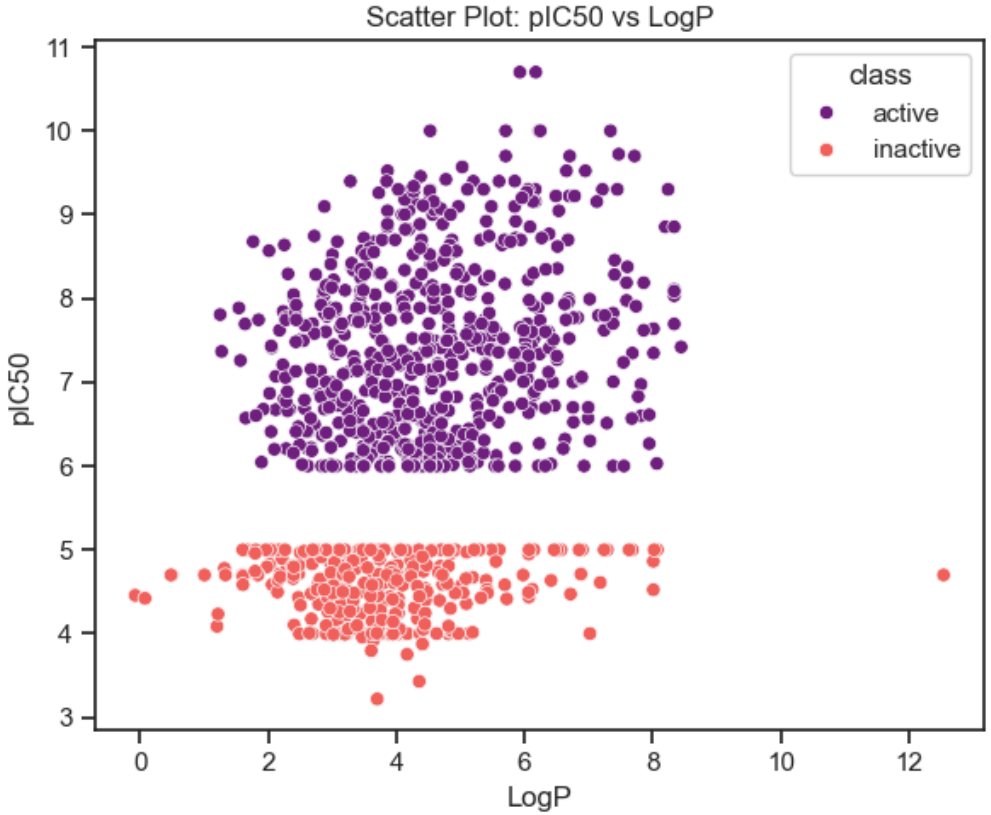

MW
LogP
NumHDonors
NumHAcceptors
class
active
inactive



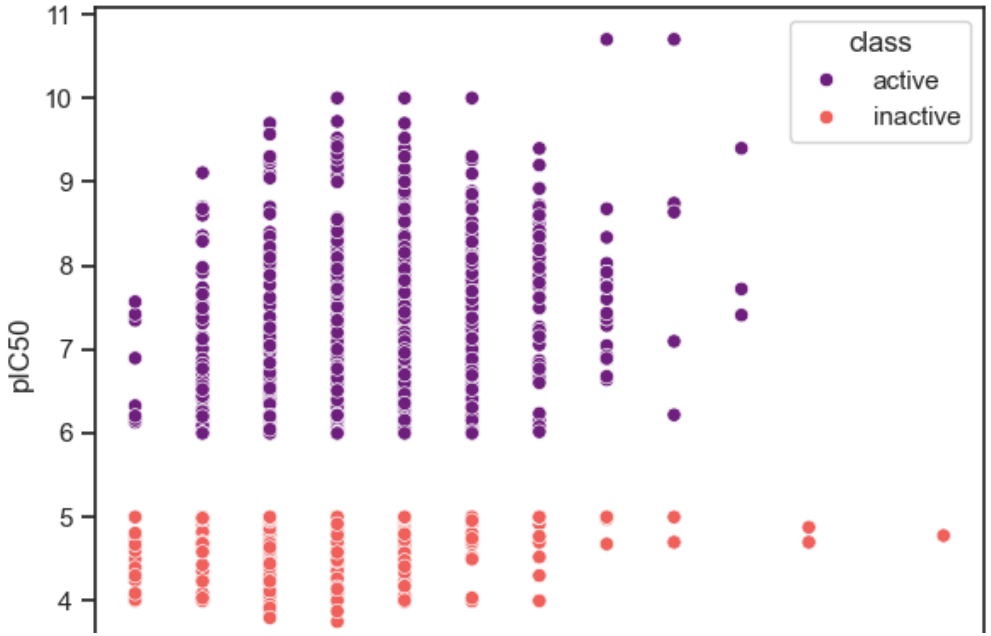
In [253…

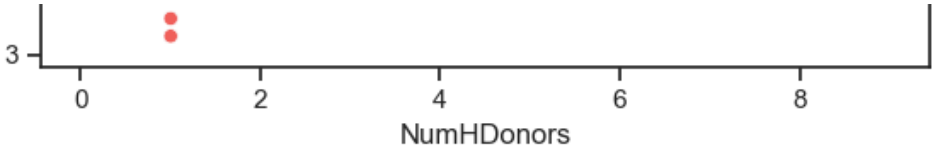
```
numeric_columns = ['MW', 'LogP', 'NumHDonors', 'NumHAcceptors']
# Set figure size
plt.figure(figsize=(12, 10))
# Loop through numeric columns and create scatter plots
for i, column in enumerate(numeric_columns, 1):
 plt.subplot(2, 2, i) # 2 rows, 2 columns layout
 sns.scatterplot(x=df_2class[column], y=df_2class['pIC50'], hue=df_2class['class'], palette='magma')
 plt.title(f'Scatter Plot: pIC50 vs {column}')
 plt.xlabel(column)
 plt.ylabel('pIC50')
# Adjust layout
plt.tight_layout()
plt.show()
```

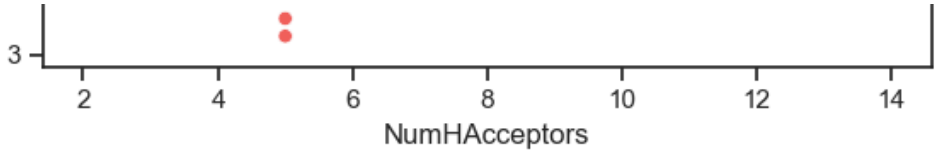

Scatter Plot: pIC50 vs MW
class
active
inactive
pIC50
MW
11
10
9
8
7
6
5
4
3
200
300
400
500
600
700
800


Scatter Plot: pIC50 vs LogP
class
active
inactive
pIC50
LogP
11
10
9
8
7
6
5
4
3
0
2
4
6
8
10
12


Scatter Plot: pIC50 vs NumHDonors
class
active
inactive
pIC50
11
10
9
8
7
6
5
4

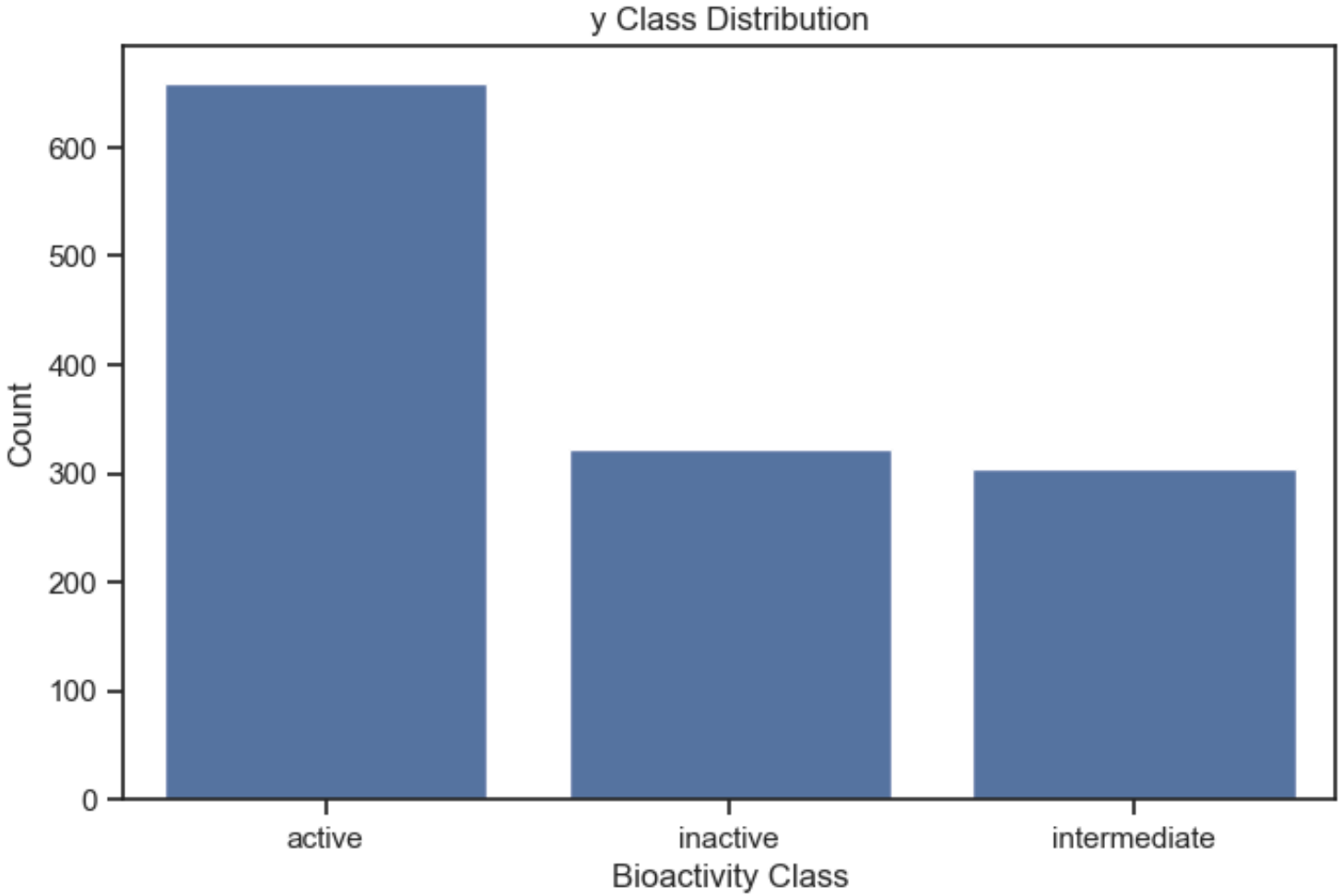

Scatter Plot: pIC50 vs NumHAcceptors
class
active
inactive
pIC50
11
10
9
8
7
6
5
4



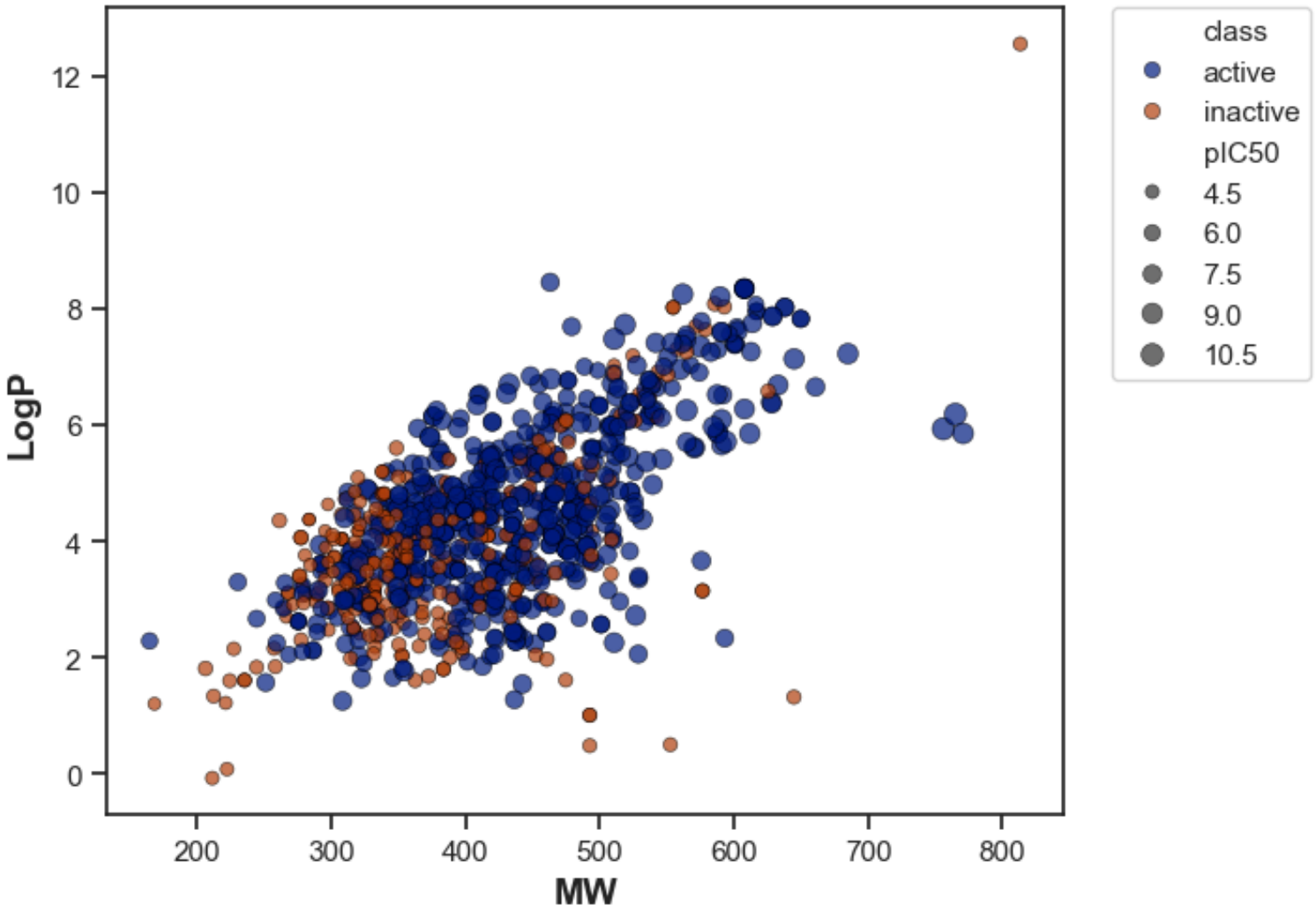


In [257…

```
plt.figure(figsize=(8, 5))
sns.countplot(x="class", data=df, )
plt.xlabel("Bioactivity Class")
plt.ylabel("Count")
plt.title("y Class Distribution")
plt.show()
```

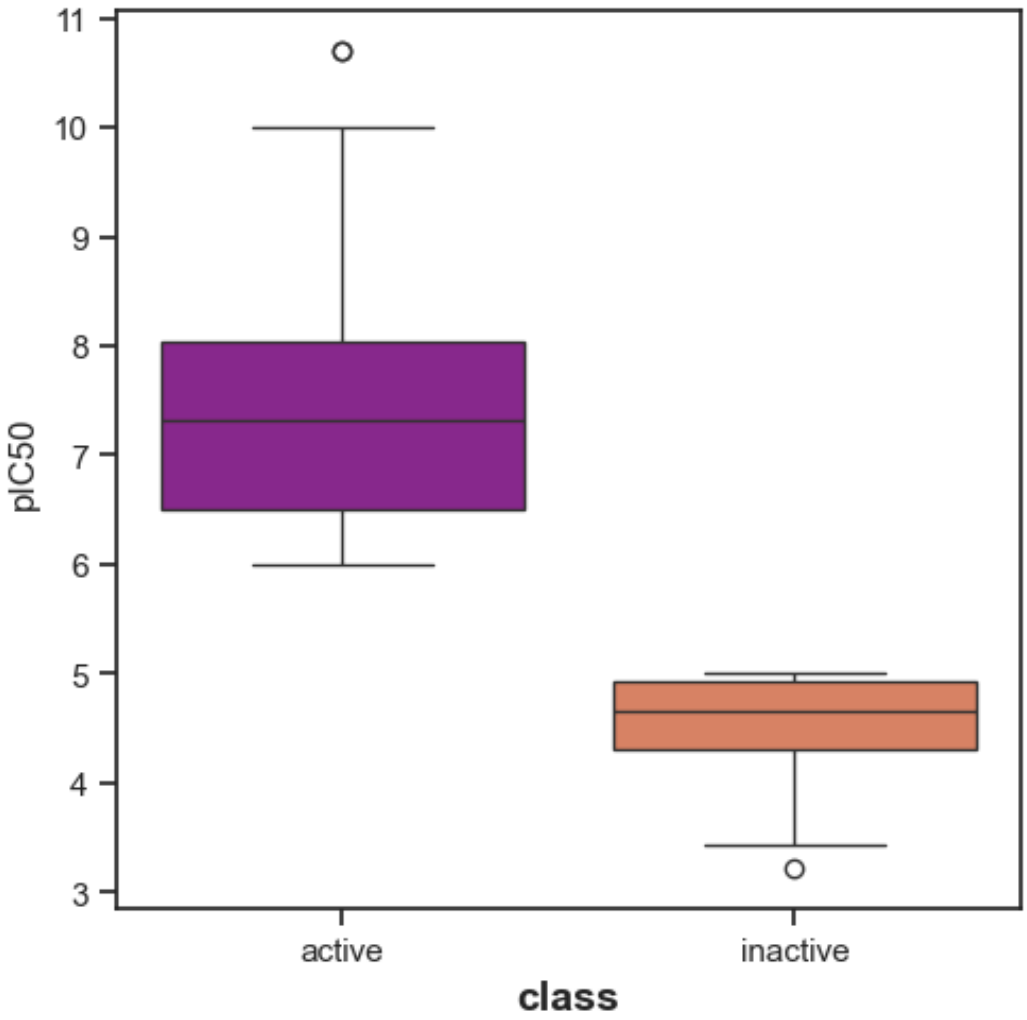


In [261…]
```
plt.figure(figsize=(6.6, 5.6))
sns.scatterplot(x='MW', y='LogP', data=df_2class, hue='class', size='pIC50',palette='dark', edgecolor='black', alpha=0.7)
plt.xlabel('MW', fontsize=14, fontweight='bold')
plt.ylabel('LogP', fontsize=14, fontweight='bold')
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0)
plt.show()
```



In [266…]
```
plt.figure(figsize=(5.5, 5.5))
sns.boxplot(x = 'class', y = 'pIC50', data = df_2class, hue = 'class' ,palette = 'plasma')
plt.xlabel('class', fontsize=14, fontweight='bold')
```

Out[266… Text(0.5, 0, 'class')



Statistical analysis | Mann-Whitney U Test

In [274…

```
 # https://machinelearningmastery.com/nonparametric-statistical-significance-tests-in-python/
from numpy.random import seed
from scipy.stats import mannwhitneyu

def mannwhitney(descriptor, verbose=False):
    # Seed the random number generator for reproducibility
    seed(1)
```

```
    # Select relevant columns
    selection = [descriptor, 'class']
    df = df_2class[selection]

    # Split active and inactive classes
    active = df[df['class'] == 'active'][descriptor]
    inactive = df[df['class'] == 'inactive'][descriptor]

    # Perform Mann-Whitney U test
    stat, p = mannwhitneyu(active, inactive)

    # Interpret the result
    alpha = 0.05
    interpretation = 'Same distribution (fail to reject H0)' if p > alpha else 'Different distribution (reject H0)'

    # Store results in a DataFrame
    results = pd.DataFrame({
        'Descriptor': [descriptor],
        'Statistics': [stat],
        'p': [p],
        'alpha': [alpha],
        'Interpretation': [interpretation]
    })

    # Save results to CSV
    filename = f'mannwhitneyu_{descriptor}.csv'
    results.to_csv(filename, index=False)

    return results
```

In [277…

```
mannwhitney('pIC50')
```

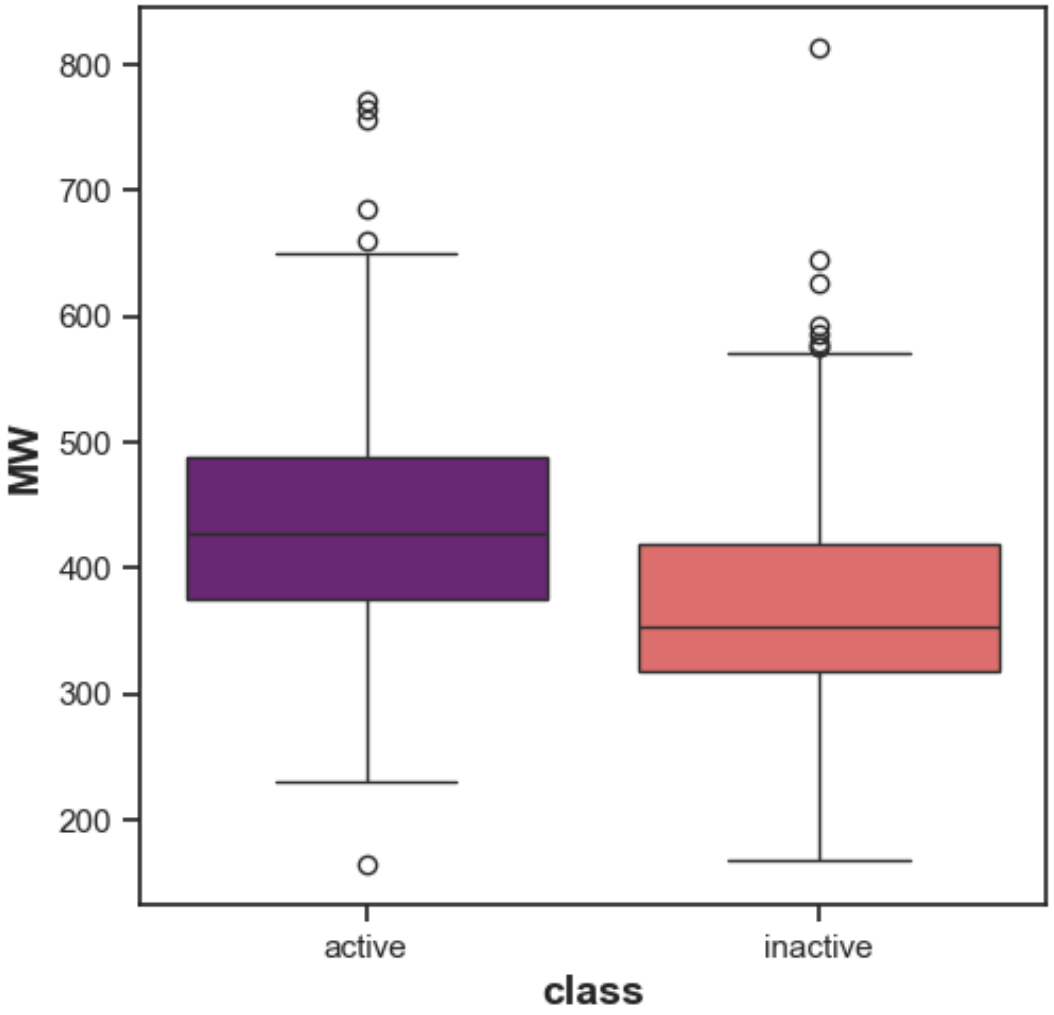
Out[277…

	Descriptor	Statistics	p	alpha	Interpretation
0	pIC50	213180.0	2.036349e-143	0.05	Different distribution (reject H0)

MW

In [292…
```
plt.figure(figsize=(5.5, 5.5))
sns.boxplot(x = 'class', y = 'MW', data = df_2class,hue = 'class' , palette = 'magma')
plt.xlabel('class', fontsize=14, fontweight='bold')
plt.ylabel('MW', fontsize=14, fontweight='bold')
```

Out[292…
```
Text(0, 0.5, 'MW')
```



Statistical analysis | Mann-Whitney U Test

In [297…
```
mannwhitney('MW')
```

Out[297…

	Descriptor	Statistics	p	alpha	Interpretation
0	MW	151302.0	1.086061e-26	0.05	Different distribution (reject H0)

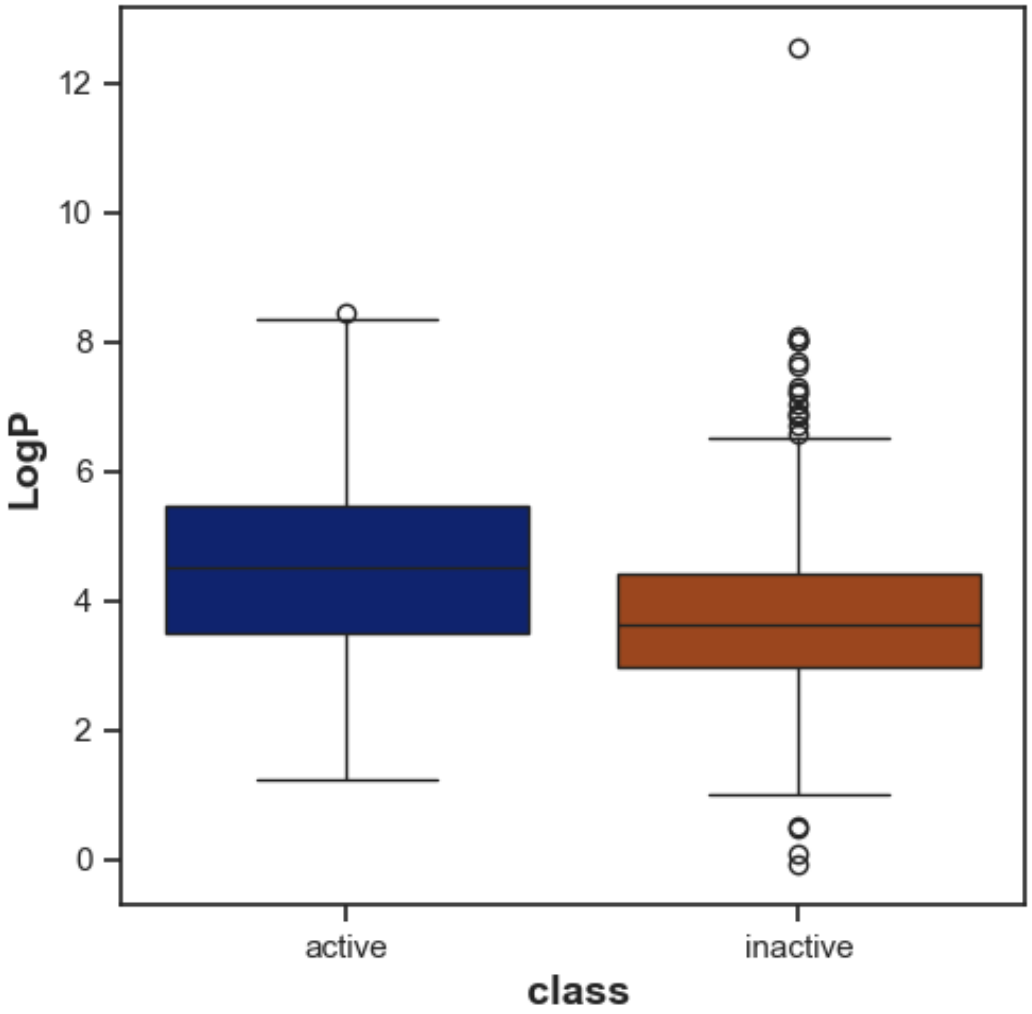
LogP

In [303…

```
plt.figure(figsize=(5.5, 5.5))

sns.boxplot(x = 'class', y = 'LogP', data = df_2class,hue = 'class' , palette = 'dark')

plt.xlabel(' class', fontsize=14, fontweight='bold')
plt.ylabel('LogP', fontsize=14, fontweight='bold')
```

Out[303…

```
Text(0, 0.5, 'LogP')
```



Statistical analysis | Mann-Whitney U Test

In [307…
```
mannwhitney('LogP')
```

Out[307…

	Descriptor	Statistics	p	alpha	Interpretation
0	LogP	141711.0	4.463653e-17	0.05	Different distribution (reject H0)

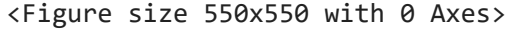
NumHDonors

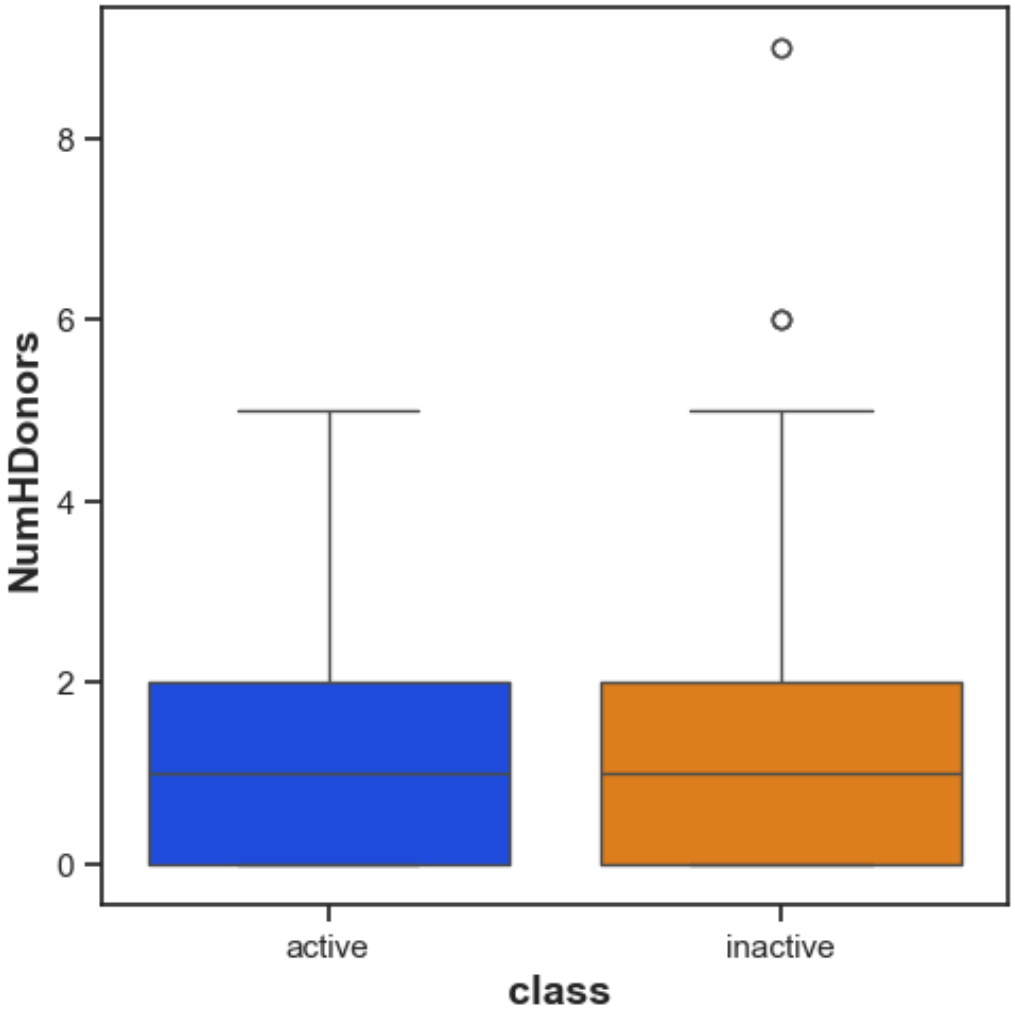
In [315…

```
plt.figure(figsize=(5.5, 5.5))
plt.figure(figsize=(5.5, 5.5))
sns.boxplot(x = 'class', y = 'NumHDonors', data = df_2class,hue = 'class' , palette = 'bright')
plt.xlabel(' class', fontsize=14, fontweight='bold')
plt.ylabel('NumHDonors', fontsize=14, fontweight='bold')
```

Out[315…

```
Text(0, 0.5, 'NumHDonors')
```





Statistical analysis | Mann-Whitney U Test

In [323…
```
mannwhitney('NumHDonors')
```

Out[323…

	Descriptor	Statistics	p	alpha	Interpretation
0	NumHDonors	111706.5	0.196062	0.05	Same distribution (fail to reject H0)

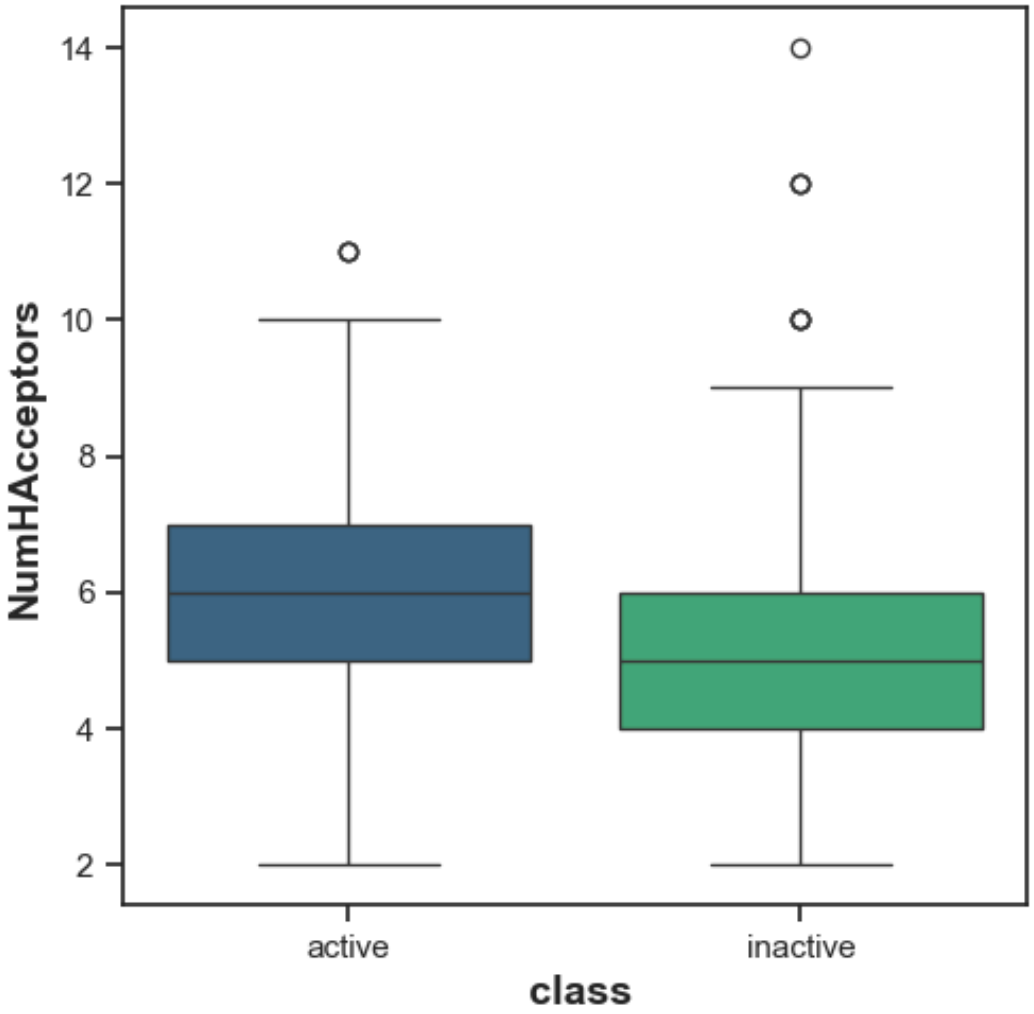
NumHAcceptors

In [329…

```
plt.figure(figsize=(5.5, 5.5))

sns.boxplot(x = 'class', y = 'NumHAcceptors', data = df_2class,hue = 'class' , palette = 'viridis')

plt.xlabel(' class', fontsize=14, fontweight='bold')
plt.ylabel('NumHAcceptors', fontsize=14, fontweight='bold')
```

Out[329…

```
Text(0, 0.5, 'NumHAcceptors')
```



Statistical analysis | Mann-Whitney U Test

In [332...
```
mannwhitney('NumHAcceptors')
```

Out[332...

	Descriptor	Statistics	p	alpha	Interpretation
0	NumHAcceptors	124596.0	0.000012	0.05	Different distribution (reject H0)

Box Plots pIC50 values Taking a look at pIC50 values, the actives and inactives displayed statistically significant difference, which is to be expected since threshold values (IC50 < 1,000 nM = Actives while IC50 > 10,000 nM = Inactives, corresponding to pIC50 > 6 = Actives and pIC50 < 5 =

Inactives) were used to define actives and inactives.

Lipinski's descriptors Of the 4 Lipinski's descriptors (MW, LogP, NumHDonors and NumHAcceptors), only MW, LogP,and NumHAcceptors exhibited difference between the actives and inactives while the other descriptors (only NumHDonors) shows statistically significant same difference between actives and inactives.

In [345…
```
df3 = pd.read_csv(r"C:\Users\manoj\OneDrive\Desktop\Phosphodiesterase 4D\Phosphodiesterase 4D_04_bioactivity_data_3class_pIC50.csv"
```

In [347…
```
df3
```

Out[347…

	Unnamed: 0	molecule_chembl_id	canonical_smiles	class	MW	LogP	NumHDonors	NumHAcceptors
0	0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	active	343.423	4.05278	1.0	4.0
1	1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	active	371.352	4.57020	1.0	5.0
2	2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	active	368.352	4.19820	1.0	6.0
3	3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	active	464.562	4.81650	1.0	7.0
4	4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	inactive	494.588	4.17900	2.0	8.0
...	...	...	...	...	...	...	...	...
978	1282	CHEMBL5405731	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0
979	1283	CHEMBL5435764	COc1c(OCc2cccc(F)c2)cc2oc3cc4c(c(OCc5ccccn5)c3...	active	607.678	8.33980	0.0	7.0
980	1284	CHEMBL5421333	COc1c(OCc2ccccn2)cc2oc3cc4c(c(OCc5ccccn5)c3c(=...	active	590.676	7.59570	0.0	8.0
981	1285	CHEMBL5440771	COc1c(OCc2ccc(F)cc2)cc2oc3cc4c(c(OCc5cccnc5)c3...	active	607.678	8.33980	0.0	7.0
982	1286	CHEMBL5417689	CCOc1cc([C@H](Cc2c(Cl)c[n+]([O-])cc2Cl)OC(=O)c...	active	770.732	5.85120	2.0	11.0

983 rows × 9 columns

In [350…
```
df3.head()
```

Out[350…

	Unnamed: 0	molecule_chembl_id	canonical_smiles	class	MW	LogP	NumHDonors	NumHAcceptors	pIC5
0	0	CHEMBL511115	COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1	active	343.423	4.05278	1.0	4.0	7.2006
1	1	CHEMBL74078	O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-]...	active	371.352	4.57020	1.0	5.0	9.0000(
2	2	CHEMBL77826	O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1	active	368.352	4.19820	1.0	6.0	8.8239(
3	3	CHEMBL97817	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(O...	active	464.562	4.81650	1.0	7.0	7.6575
4	4	CHEMBL319809	CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c...	inactive	494.588	4.17900	2.0	8.0	4.0000(

In [352…
```
df3.describe()
```

Out[352…

	Unnamed: 0	MW	LogP	NumHDonors	NumHAcceptors	pIC50
count	983.000000	983.000000	983.000000	983.000000	983.000000	983.000000
mean	636.004069	416.104192	4.320447	1.073245	5.595117	6.459856
std	376.959983	90.827516	1.502245	1.117682	1.768374	1.577922
min	0.000000	164.595000	-0.074800	0.000000	2.000000	3.219827
25%	302.500000	350.075000	3.280450	0.000000	4.000000	4.958607
50%	639.000000	410.436000	4.173500	1.000000	6.000000	6.514279
75%	961.500000	475.372500	5.190250	2.000000	7.000000	7.657577
max	1286.000000	813.302000	12.544800	9.000000	14.000000	10.700057

In [368…
```
selection = ['canonical_smiles','molecule_chembl_id']
df3_selection = df3[selection]
df3_selection.to_csv('molecule.smi', sep='\t', index=False, header=False)
```

In [370…
```
with open('molecule.smi', 'r') as file:
 for _ in range(5):
     print(file.readline().strip())
```

```
COc1ccc([C@]2(C#N)CC[C@@H](C(=O)O)CC2)cc1OC1CCCC1       CHEMBL511115
O=C(O)c1ccc(-c2cc3cccnc3c(-c3cccc([N+](=O)[O-])c3)n2)cc1        CHEMBL74078
O=C(O)c1ccc(-c2cc3cccnc3c(-c3ccc4nonc4c3)n2)cc1 CHEMBL77826
CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OC)c(OCCO)cc2n1 CHEMBL97817
CCc1nc(-c2cc(OCC3CC3)cc(OCC3CC3)c2)c2cc(OCCO)c(OCCO)cc2n1       CHEMBL319809
```

In [371…
```
with open('molecule.smi', 'r') as file:
 line_count = sum(1 for line in file)
print("Total number of lines:", line_count)
```

```
Total number of lines: 983
```

Preparing the X and Y Data Matrices

In [431…
```
X = pd.read_csv(r"C:\Users\manoj\OneDrive\Desktop\New folder\archive\descriptors_output.csv")
```

In [432…
```
X
```

Out[432...

	Name	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	Pubch
0	CHEMBL130478	1	1	0	0	0	0	0	0	
1	CHEMBL336538	1	1	1	0	0	0	0	0	
2	CHEMBL339995	1	1	1	0	0	0	0	0	
3	CHEMBL341437	1	1	1	0	0	0	0	0	
4	CHEMBL130098	1	1	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...
6941	CHEMBL253998	1	1	1	0	0	0	0	0	
6942	CHEMBL502	1	1	1	0	0	0	0	0	
6943	CHEMBL3085398	1	1	1	0	0	0	0	0	
6944	CHEMBL13045	1	1	1	0	0	0	0	0	
6945	CHEMBL417799	1	1	0	0	0	0	0	0	

6946 rows × 882 columns

In [434...

```
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6946 entries, 0 to 6945
Columns: 882 entries, Name to PubchemFP880
dtypes: int64(881), object(1)
memory usage: 46.7+ MB
```

In [436...

```
X.head()
```

Out[436…

	Name	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	PubchemFP
0	CHEMBL130478	1	1	0	0	0	0	0	0	
1	CHEMBL336538	1	1	1	0	0	0	0	0	
2	CHEMBL339995	1	1	1	0	0	0	0	0	
3	CHEMBL341437	1	1	1	0	0	0	0	0	
4	CHEMBL130098	1	1	0	0	0	0	0	0	

5 rows × 882 columns

In [438…

```
X.dtypes
```

Out[438…

```
Name            object
PubchemFP0       int64
PubchemFP1       int64
PubchemFP2       int64
PubchemFP3       int64
                 ...
PubchemFP876     int64
PubchemFP877     int64
PubchemFP878     int64
PubchemFP879     int64
PubchemFP880     int64
Length: 882, dtype: object
```

In [440…

```
X = X.drop(columns=['Name'])
X
```

Out[440...

	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	PubchemFP8	PubchemI
0	1	1	0	0	0	0	0	0	0	
1	1	1	1	0	0	0	0	0	0	
2	1	1	1	0	0	0	0	0	0	
3	1	1	1	0	0	0	0	0	0	
4	1	1	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	
6941	1	1	1	0	0	0	0	0	0	
6942	1	1	1	0	0	0	0	0	0	
6943	1	1	1	0	0	0	0	0	0	
6944	1	1	1	0	0	0	0	0	0	
6945	1	1	0	0	0	0	0	0	0	

6946 rows × 881 columns

Y variable

In [444...

```
y = df3['class']
```

In [445...

```
y = y.map({'active': 1, 'inactive': 0})
```

Split dataset

In [449...

```
print("Shape of X:", X.shape)
print("Shape of y:", y.shape)
```

```
Shape of X: (6946, 881)
Shape of y: (983,)
```

In [454…
```
X = X.iloc[:y.shape[0], :]  # Trim X to match y
```

In [457…
```
print("Shape of X:", X.shape)
print("Shape of y:", y.shape)
```

```
Shape of X: (983, 881)
Shape of y: (983,)
```

In [461…
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

In [464…
```
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

In [467…
```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Logistic Regression Model

In [470…
```
lr = LogisticRegression(max_iter=600)
```

In [473…
```
lr.fit(X_train, y_train)
```

Out[473…

▾ LogisticRegression

```
LogisticRegression(max_iter=600)
```

In [476…
```
y_pred_lr = lr.predict(X_test)
y_pred_lr
```

Out[476…
```
array([0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0,
       1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1,
       1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0,
       0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0,
       0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 1], dtype=int64)
```

In [479…
```
print(classification_report(y_test,y_pred_lr))
```

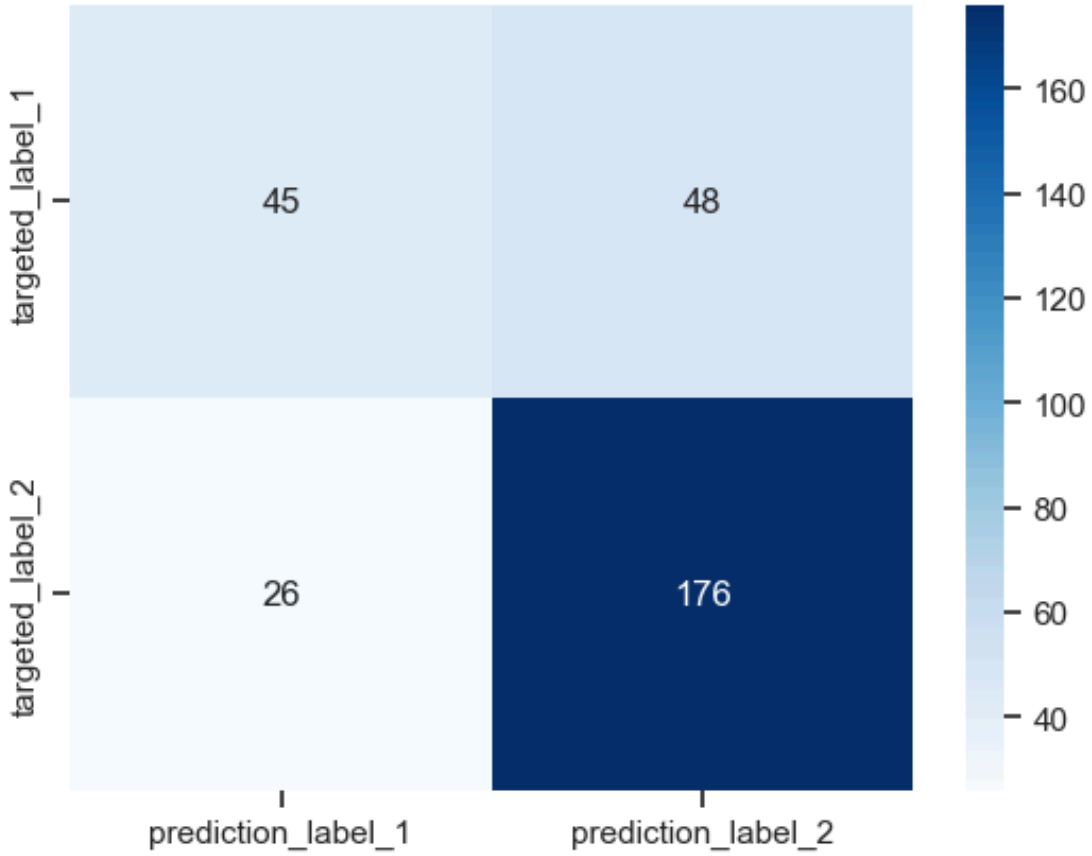
```
              precision    recall  f1-score   support

           0       0.63      0.48      0.55        93
           1       0.79      0.87      0.83       202

    accuracy                           0.75       295
   macro avg       0.71      0.68      0.69       295
weighted avg       0.74      0.75      0.74       295
```

In [484…
```
accuracy = accuracy_score(y_test, y_pred_lr)
print(f"Logistic Regression Model Accuracy: {accuracy * 100:.2f}%")
```

```
Logistic Regression Model Accuracy: 74.92%
```

In [491…
```
cm = confusion_matrix(y_test, y_pred_lr)
ax = sns.heatmap(cm, annot=True, fmt='d', cmap="Blues")
ax.xaxis.set_ticklabels(['prediction_label_1', 'prediction_label_2' ])
ax.yaxis.set_ticklabels(['targeted_label_1', 'targeted_label_2' ])
```

Out[491…
```
[Text(0, 0.5, 'targeted_label_1'), Text(0, 1.5, 'targeted_label_2')]
```



Logistic Regression Model Accuracy: 74.92%

In [498…
```
from sklearn.tree import DecisionTreeClassifier
DT = DecisionTreeClassifier()
DT.fit(X_train,y_train)
```

Out[498…
```
DecisionTreeClassifier()
```

In [501…
```
y_pred_DT = DT.predict(X_test)
y_pred_DT
```

Out[501…

```
array([1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0,
       1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1,
       0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0,
       1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0,
       1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
       0, 1, 1, 1, 1, 1, 1, 0, 1], dtype=int64)
```

In [504…

```
print(classification_report(y_test,y_pred_DT))
```

```
              precision    recall  f1-score   support

           0       0.56      0.61      0.59        93
           1       0.81      0.78      0.80       202

    accuracy                           0.73       295
   macro avg       0.69      0.70      0.69       295
weighted avg       0.74      0.73      0.73       295
```

In [539…

```
accuracy = accuracy_score(y_test, y_pred_rnf)
print(f"RandomForestClassifier Model Accuracy: {accuracy * 100:.2f}%")
```

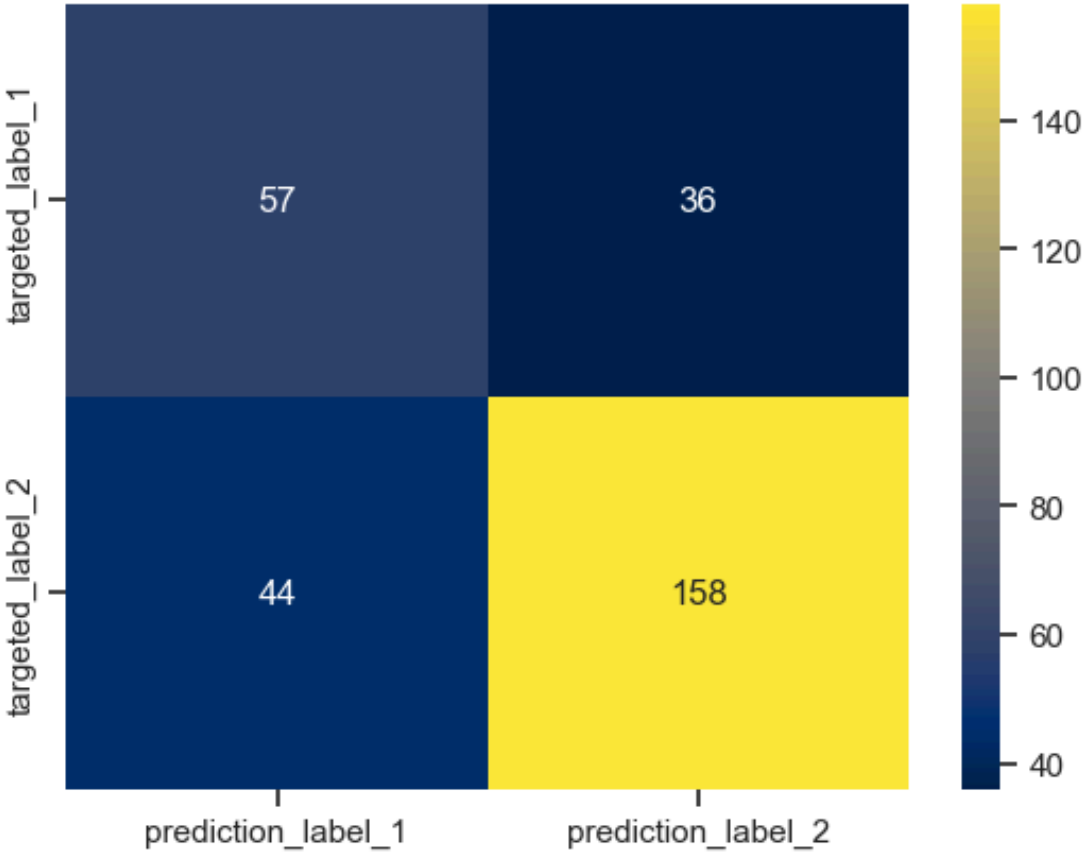
```
RandomForestClassifier Model Accuracy: 74.24%
```

In [510…

```
cm = confusion_matrix(y_test, y_pred_DT)
ax = sns.heatmap(cm, annot=True, fmt='d', cmap="cividis")
ax.xaxis.set_ticklabels(['prediction_label_1', 'prediction_label_2' ])
ax.yaxis.set_ticklabels(['targeted_label_1', 'targeted_label_2' ])
```

Out[510…

```
[Text(0, 0.5, 'targeted_label_1'), Text(0, 1.5, 'targeted_label_2')]
```



DecisionTreeClassifier Model Accuracy: 72.88%

RandomForestClassifier

In [524…
```
rnf = RandomForestClassifier()
rnf.fit(X_train,y_train)
```

Out[524…
```
RandomForestClassifier
RandomForestClassifier()
```

In [525…
```
y_pred_rnf = rnf.predict(X_test)
```

```
y_pred_rnf
```

Out[525…

```
array([0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
       1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0,
       0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0,
       0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 1], dtype=int64)
```

In [527…

```
print(classification_report(y_test,y_pred_rnf))
```

```
              precision    recall  f1-score   support

           0       0.60      0.54      0.57        93
           1       0.80      0.84      0.82       202

    accuracy                           0.74       295
   macro avg       0.70      0.69      0.69       295
weighted avg       0.74      0.74      0.74       295
```

In [534…

```
accuracy = accuracy_score(y_test, y_pred_rnf)
print(f"DecisionTreeClassifier Model Accuracy: {accuracy * 100:.2f}%")
```

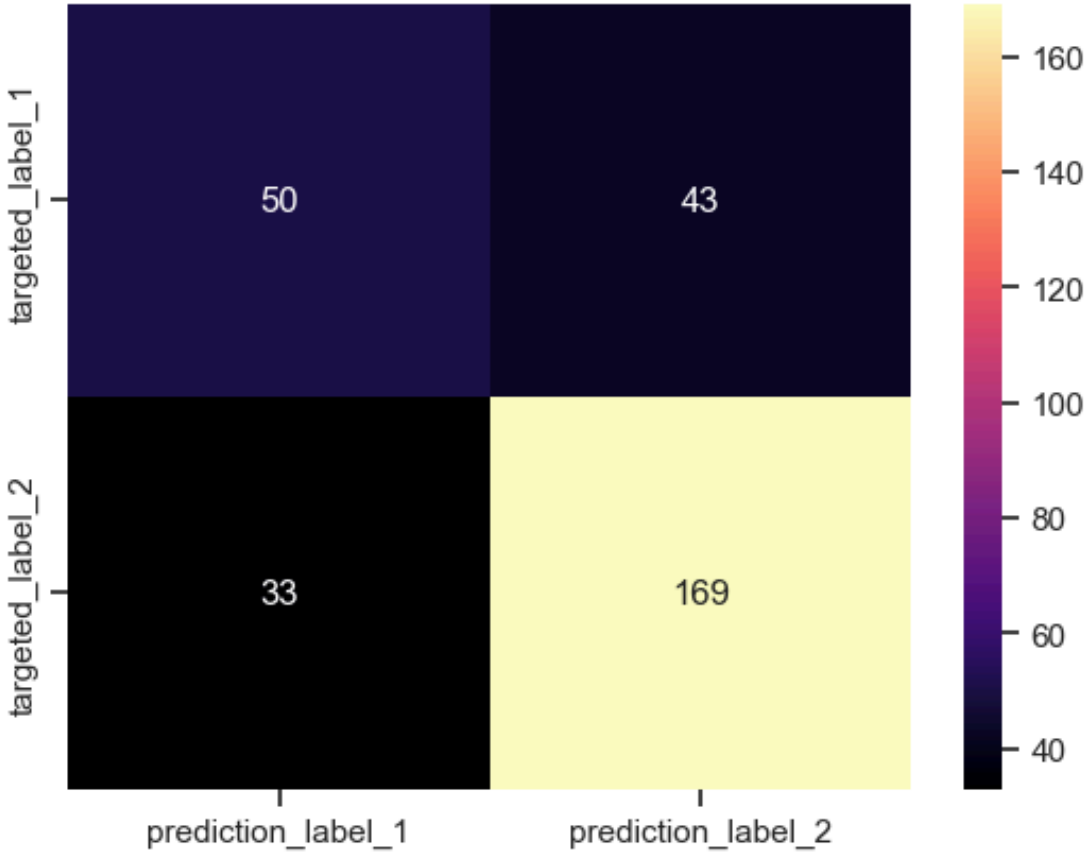
```
DecisionTreeClassifier Model Accuracy: 74.24%
```

In [531…

```
cm = confusion_matrix(y_test, y_pred_rnf)
ax = sns.heatmap(cm, annot=True, fmt='d', cmap="magma")
ax.xaxis.set_ticklabels(['prediction_label_1', 'prediction_label_2' ])
ax.yaxis.set_ticklabels(['targeted_label_1', 'targeted_label_2' ])
```

Out[531…

```
[Text(0, 0.5, 'targeted_label_1'), Text(0, 1.5, 'targeted_label_2')]
```



DecisionTreeClassifier Model Accuracy: 74.24%

K-Nearest Neighbors (KNN)

In [549…

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train,y_train)
```

Out[549…

KNeighborsClassifier

```
KNeighborsClassifier()
```

In [551…
```
y_pred_knn = knn.predict(X_test)
y_pred_knn
```

Out[551…
```
array([0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
       0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
       1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 1, 1], dtype=int64)
```

In [552…
```
print(classification_report(y_test,y_pred_knn))
```

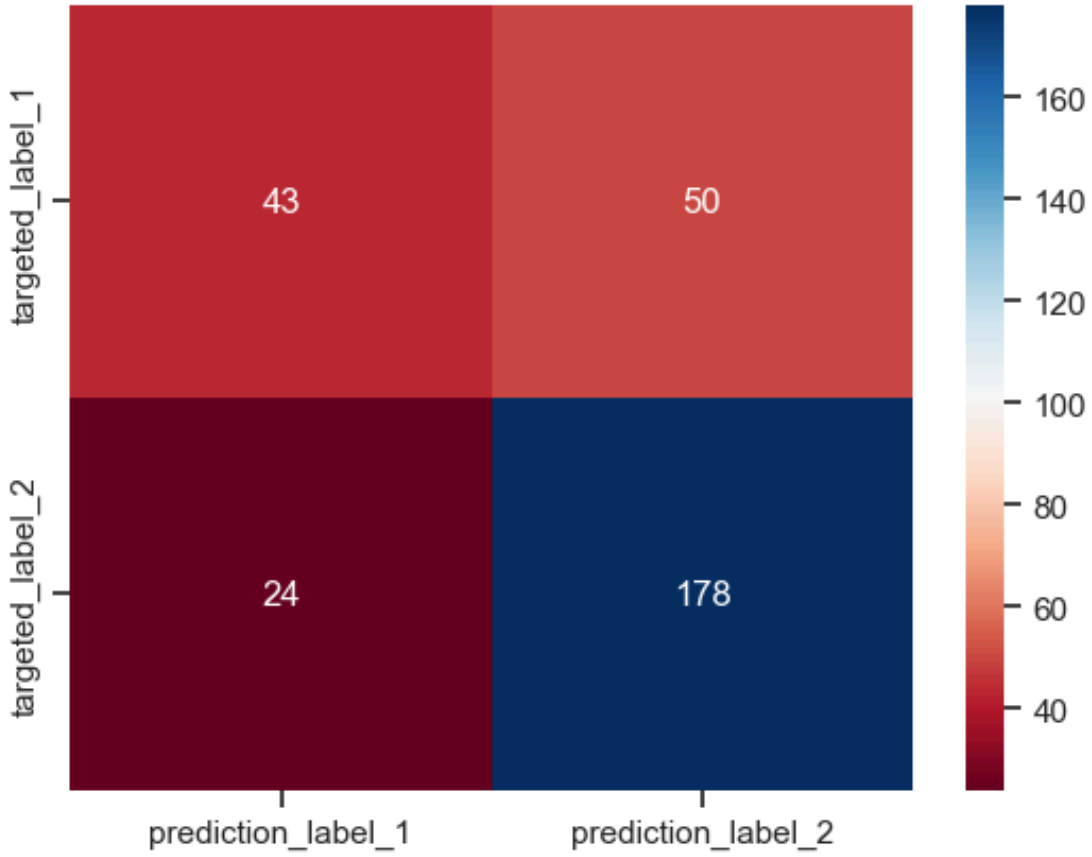
```
              precision    recall  f1-score   support

           0       0.64      0.46      0.54        93
           1       0.78      0.88      0.83       202

    accuracy                           0.75       295
   macro avg       0.71      0.67      0.68       295
weighted avg       0.74      0.75      0.74       295
```

In [556…
```
accuracy = accuracy_score(y_test, y_pred_knn)
print(f"KNeighborsClassifier Model Accuracy: {accuracy * 100:.2f}%")
```

```
KNeighborsClassifier Model Accuracy: 74.92%
```

In [573…
```
cm = confusion_matrix(y_test, y_pred_knn)
ax = sns.heatmap(cm, annot=True, fmt='d', cmap="RdBu")
ax.xaxis.set_ticklabels(['prediction_label_1', 'prediction_label_2' ])
ax.yaxis.set_ticklabels(['targeted_label_1', 'targeted_label_2' ])
```

Out[573…
```
[Text(0, 0.5, 'targeted_label_1'), Text(0, 1.5, 'targeted_label_2')]
```



KNeighborsClassifier Model Accuracy: 74.92%

In [581…
```
df3_X = pd.read_csv(r"C:\Users\manoj\OneDrive\Desktop\New folder\archive\descriptors_output.csv")
```

In [585…
```
df3_X
```

Out[585...

	Name	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	Pubch
0	CHEMBL130478	1	1	0	0	0	0	0	0	
1	CHEMBL336538	1	1	1	0	0	0	0	0	
2	CHEMBL339995	1	1	1	0	0	0	0	0	
3	CHEMBL341437	1	1	1	0	0	0	0	0	
4	CHEMBL130098	1	1	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...
6941	CHEMBL253998	1	1	1	0	0	0	0	0	
6942	CHEMBL502	1	1	1	0	0	0	0	0	
6943	CHEMBL3085398	1	1	1	0	0	0	0	0	
6944	CHEMBL13045	1	1	1	0	0	0	0	0	
6945	CHEMBL417799	1	1	0	0	0	0	0	0	

6946 rows × 882 columns

In [588...

```
df3_X = df3_X.drop(columns=['Name'])
df3_X
```

Out[588...

	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	PubchemFP8	PubchemI
0	1	1	0	0	0	0	0	0	0	
1	1	1	1	0	0	0	0	0	0	
2	1	1	1	0	0	0	0	0	0	
3	1	1	1	0	0	0	0	0	0	
4	1	1	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	
6941	1	1	1	0	0	0	0	0	0	
6942	1	1	1	0	0	0	0	0	0	
6943	1	1	1	0	0	0	0	0	0	
6944	1	1	1	0	0	0	0	0	0	
6945	1	1	0	0	0	0	0	0	0	

6946 rows × 881 columns

In [615...

```
Y = df3['pIC50']
Y
```

Out[615…

```
0      7.200659
1      9.000000
2      8.823909
3      7.657577
4      4.000000
         ...
978    7.694649
979    8.031517
980    8.187087
981    8.086186
982    9.400008
Name: pIC50, Length: 983, dtype: float64
```

In [617…

```
dataset3 = pd.concat([df3_X,df3_Y], axis=1)
dataset3
```

Out[617…

	PubchemFP0	PubchemFP1	PubchemFP2	PubchemFP3	PubchemFP4	PubchemFP5	PubchemFP6	PubchemFP7	PubchemFP8	PubchemI
0	1	1	0	0	0	0	0	0	0	
1	1	1	1	0	0	0	0	0	0	
2	1	1	1	0	0	0	0	0	0	
3	1	1	1	0	0	0	0	0	0	
4	1	1	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	
6941	1	1	1	0	0	0	0	0	0	
6942	1	1	1	0	0	0	0	0	0	
6943	1	1	1	0	0	0	0	0	0	
6944	1	1	1	0	0	0	0	0	0	
6945	1	1	0	0	0	0	0	0	0	

6946 rows × 882 columns

In [619…
```
from sklearn.model_selection import train_test_split
import lazypredict
from lazypredict.Supervised import LazyRegressor
```

In [620…
```
X.shape
```

Out[620…
```
(983, 133)
```

In [621…
```
# Remove low variance features
from sklearn.feature_selection import VarianceThreshold
selection = VarianceThreshold(threshold=(.8 * (1 - .8)))
X = selection.fit_transform(X)
X.shape
```

Out[621…
```
(983, 133)
```

In [623…
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

In [626…
```
clf = LazyRegressor(verbose=0,ignore_warnings=True, custom_metric=None)
models_train,predictions_train = clf.fit(X_train, X_train, Y_train, Y_train)
models_test,predictions_test = clf.fit(X_train, X_test, Y_train, Y_test)
```

```
98%|█████████▊| 41/42 [00:37<00:00,  1.72it/s]
```

```
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhead of testing was 0.002744 seconds.
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 399
[LightGBM] [Info] Number of data points in the train set: 786, number of used features: 133
[LightGBM] [Info] Start training from score 6.463467
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
100%|██████████| 42/42 [00:37<00:00,  1.11it/s]
'tuple' object has no attribute '__name__'
Invalid Regressor(s)
 98%|█████████▊| 41/42 [00:33<00:00,  2.82it/s]
```

```
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.001508 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 399
[LightGBM] [Info] Number of data points in the train set: 786, number of used features: 133
[LightGBM] [Info] Start training from score 6.463467
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
100%|██████████| 42/42 [00:34<00:00,  1.23it/s]
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

In [631…

```
predictions_train
```

Out[631...

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
DecisionTreeRegressor	0.52	0.60	0.99	0.07
ExtraTreeRegressor	0.52	0.60	0.99	0.07
ExtraTreesRegressor	0.52	0.60	0.99	2.82
GaussianProcessRegressor	0.52	0.60	0.99	0.39
XGBRegressor	0.52	0.60	0.99	1.43
RandomForestRegressor	0.46	0.55	1.04	2.37
BaggingRegressor	0.44	0.53	1.06	0.28
MLPRegressor	0.37	0.47	1.13	4.48
HistGradientBoostingRegressor	0.36	0.47	1.13	1.98
LGBMRegressor	0.34	0.45	1.16	0.28
GradientBoostingRegressor	0.23	0.36	1.25	0.93
KNeighborsRegressor	0.19	0.33	1.28	0.12
SVR	0.13	0.28	1.32	0.43
TransformedTargetRegressor	0.12	0.27	1.33	0.22
LinearRegression	0.12	0.27	1.33	0.27
Ridge	0.12	0.27	1.33	0.05
NuSVR	0.11	0.26	1.34	0.32
RidgeCV	0.10	0.25	1.34	0.36
HuberRegressor	0.09	0.24	1.36	0.32
SGDRegressor	0.06	0.22	1.38	0.07
PoissonRegressor	0.04	0.20	1.39	0.11

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
LinearSVR	0.03	0.19	1.40	0.26
LassoCV	0.02	0.19	1.41	7.09
ElasticNetCV	0.02	0.18	1.41	9.28
BayesianRidge	0.01	0.17	1.42	0.11
OrthogonalMatchingPursuit	-0.02	0.16	1.43	0.05
AdaBoostRegressor	-0.03	0.14	1.44	0.18
TweedieRegressor	-0.03	0.14	1.44	0.07
GammaRegressor	-0.03	0.14	1.44	0.22
LassoLarsCV	-0.04	0.14	1.45	0.18
OrthogonalMatchingPursuitCV	-0.04	0.13	1.45	0.10
LassoLarsIC	-0.05	0.13	1.45	0.27
LarsCV	-0.14	0.06	1.51	0.57
LassoLars	-0.20	0.00	1.56	0.05
Lasso	-0.20	0.00	1.56	0.04
ElasticNet	-0.20	0.00	1.56	0.04
DummyRegressor	-0.20	0.00	1.56	0.03
QuantileRegressor	-0.21	-0.00	1.56	0.37
PassiveAggressiveRegressor	-0.68	-0.39	1.84	0.05
KernelRidge	-20.62	-16.95	6.60	0.09
Lars	-14764.51	-12262.84	172.46	0.14
RANSACRegressor	-2595992882468451306700800	-2156162241234942820352000	228674996968.28	1.29

In [634…

```
predictions_test
```

Out[634...

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
PoissonRegressor	-1.64	0.15	1.52	0.10
OrthogonalMatchingPursuit	-1.64	0.15	1.52	0.04
RidgeCV	-1.64	0.15	1.52	0.26
SGDRegressor	-1.65	0.15	1.53	0.06
GradientBoostingRegressor	-1.65	0.15	1.53	0.86
LassoLarsCV	-1.67	0.14	1.53	0.18
OrthogonalMatchingPursuitCV	-1.67	0.14	1.53	0.10
ElasticNetCV	-1.68	0.14	1.53	7.87
LassoCV	-1.68	0.14	1.53	6.76
LassoLarsIC	-1.68	0.14	1.54	0.25
BayesianRidge	-1.69	0.13	1.54	0.12
NuSVR	-1.72	0.12	1.55	0.20
AdaBoostRegressor	-1.77	0.11	1.56	0.18
Ridge	-1.77	0.11	1.56	0.04
TweedieRegressor	-1.77	0.11	1.56	0.06
GammaRegressor	-1.77	0.11	1.56	0.07
SVR	-1.78	0.11	1.56	0.23
TransformedTargetRegressor	-1.83	0.09	1.58	0.23
LinearRegression	-1.83	0.09	1.58	0.26
HistGradientBoostingRegressor	-1.85	0.08	1.58	1.82
LGBMRegressor	-1.89	0.07	1.59	0.56

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
LarsCV	-1.93	0.06	1.61	0.59
MLPRegressor	-2.01	0.03	1.63	4.36
KNeighborsRegressor	-2.02	0.03	1.63	0.06
HuberRegressor	-2.06	0.02	1.64	0.31
LassoLars	-2.11	-0.00	1.65	0.04
Lasso	-2.11	-0.00	1.65	0.04
DummyRegressor	-2.11	-0.00	1.65	0.03
ElasticNet	-2.11	-0.00	1.65	0.03
QuantileRegressor	-2.12	-0.00	1.66	0.33
BaggingRegressor	-2.17	-0.02	1.67	0.27
RandomForestRegressor	-2.21	-0.03	1.68	2.29
LinearSVR	-2.29	-0.06	1.70	0.25
XGBRegressor	-2.54	-0.14	1.76	0.39
ExtraTreeRegressor	-2.67	-0.18	1.80	0.06
ExtraTreesRegressor	-2.73	-0.20	1.81	2.76
DecisionTreeRegressor	-2.77	-0.21	1.82	0.05
PassiveAggressiveRegressor	-3.18	-0.34	1.92	0.05
GaussianProcessRegressor	-15.80	-4.40	3.84	0.28
KernelRidge	-49.44	-15.21	6.66	0.07
Lars	-47726.75	-15340.06	204.82	0.13
RANSACRegressor	-92727316574507265687552.00	-29805208898948762173440.00	285493992712.45	1.35

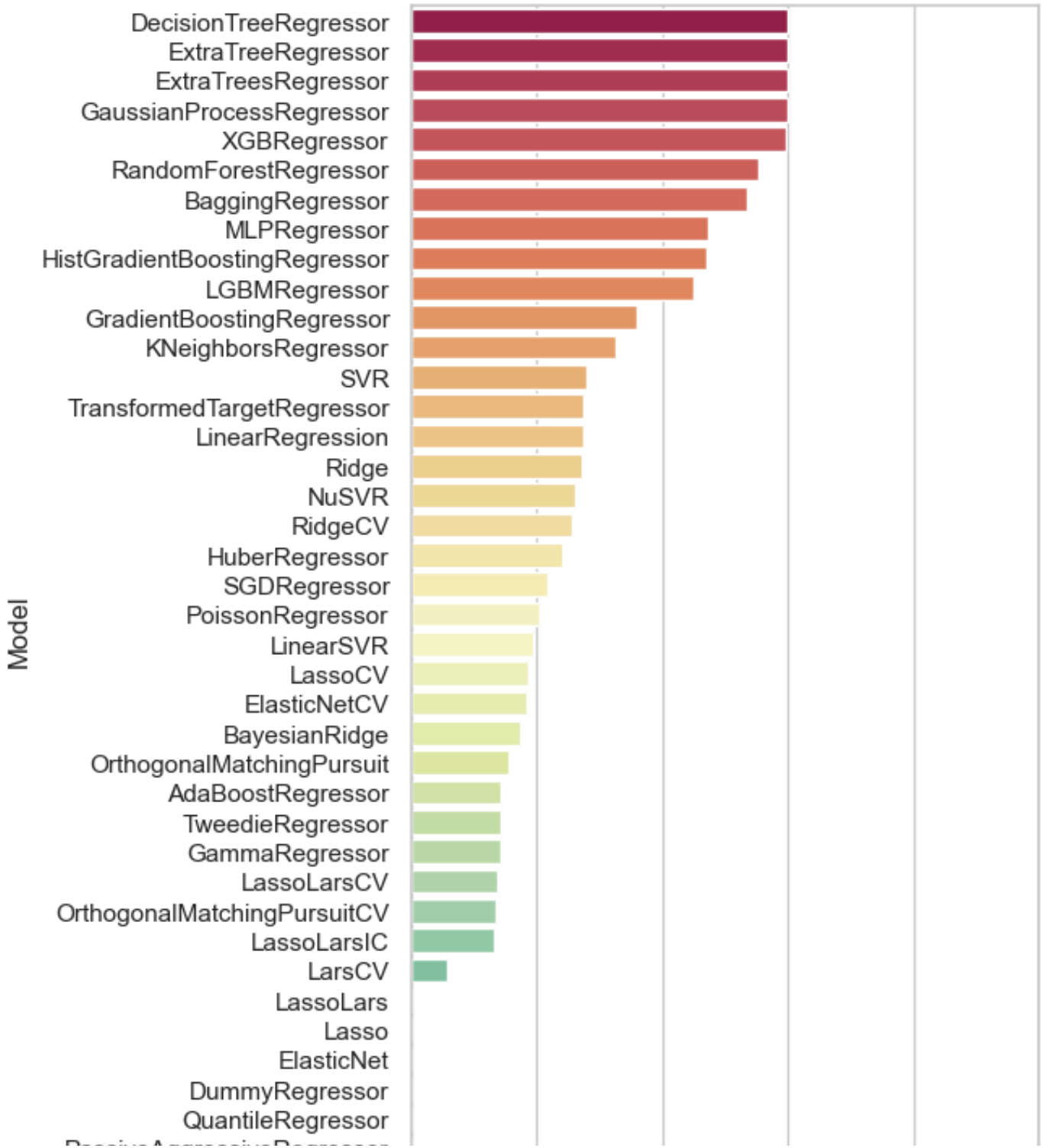
Bar plot of R-squared values

In [639…

```
#train["R-Squared"] = [0 if i < 0 else i for i in train.iloc[:,0] ]

plt.figure(figsize=(5, 10))
sns.set_theme(style="whitegrid")
ax = sns.barplot(y=predictions_train.index, x="R-Squared", data=predictions_train,palette=sns.color_palette("Spectral", len(predict
ax.set(xlim=(0, 1))
```

Out[639…

```
[(0.0, 1.0)]
```


DecisionTreeRegressor
ExtraTreeRegressor
ExtraTreesRegressor
GaussianProcessRegressor
XGBRegressor
RandomForestRegressor
BaggingRegressor
MLPRegressor
HistGradientBoostingRegressor
LGBMRegressor
GradientBoostingRegressor
KNeighborsRegressor
SVR
TransformedTargetRegressor
LinearRegression
Ridge
NuSVR
RidgeCV
HuberRegressor
SGDRegressor
PoissonRegressor
LinearSVR
LassoCV
ElasticNetCV
BayesianRidge
OrthogonalMatchingPursuit
AdaBoostRegressor
TweedieRegressor
GammaRegressor
LassoLarsCV
OrthogonalMatchingPursuitCV
LassoLarsIC
LarsCV
LassoLars
Lasso
ElasticNet
DummyRegressor
QuantileRegressor
Model

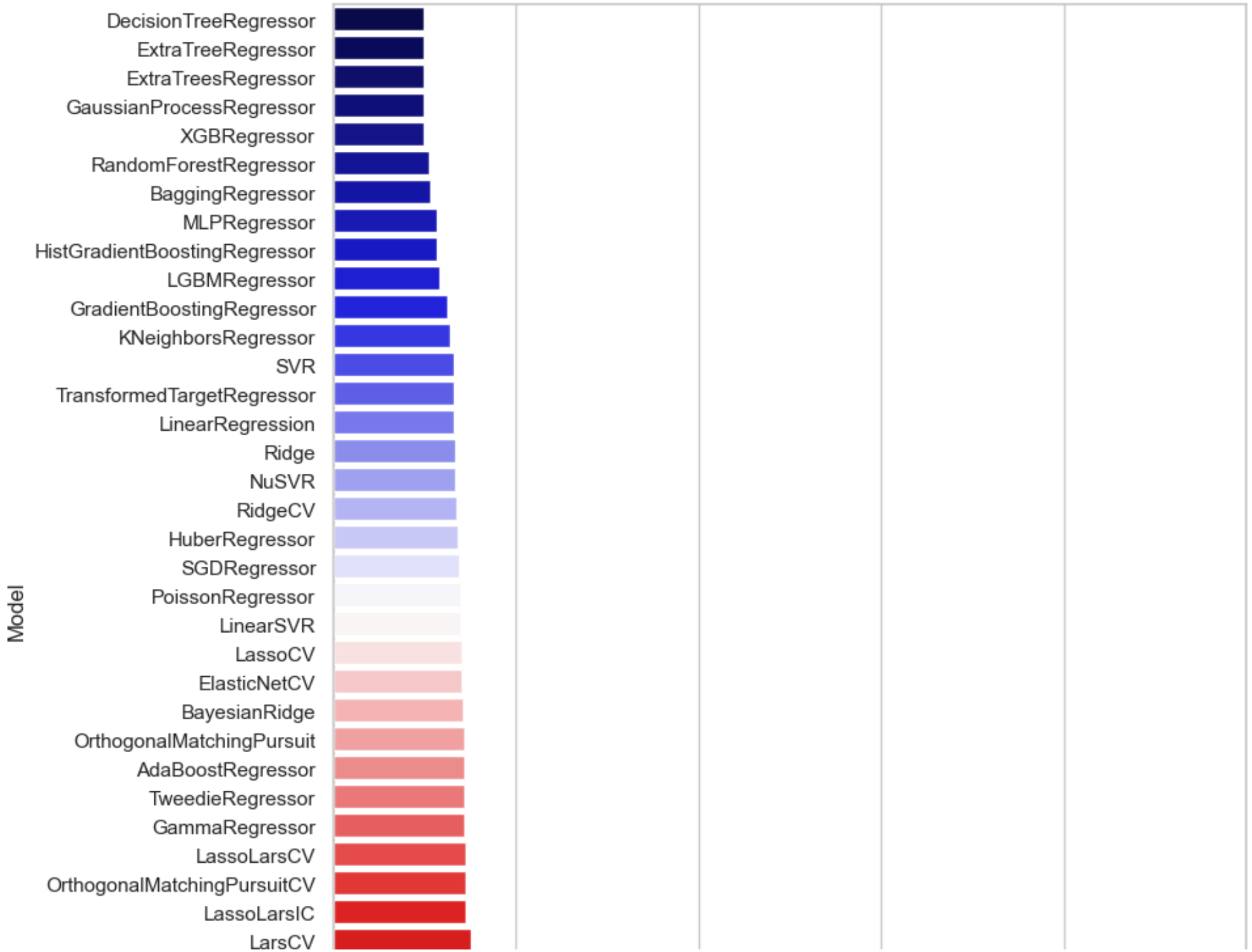
Bar plot of RMSE values

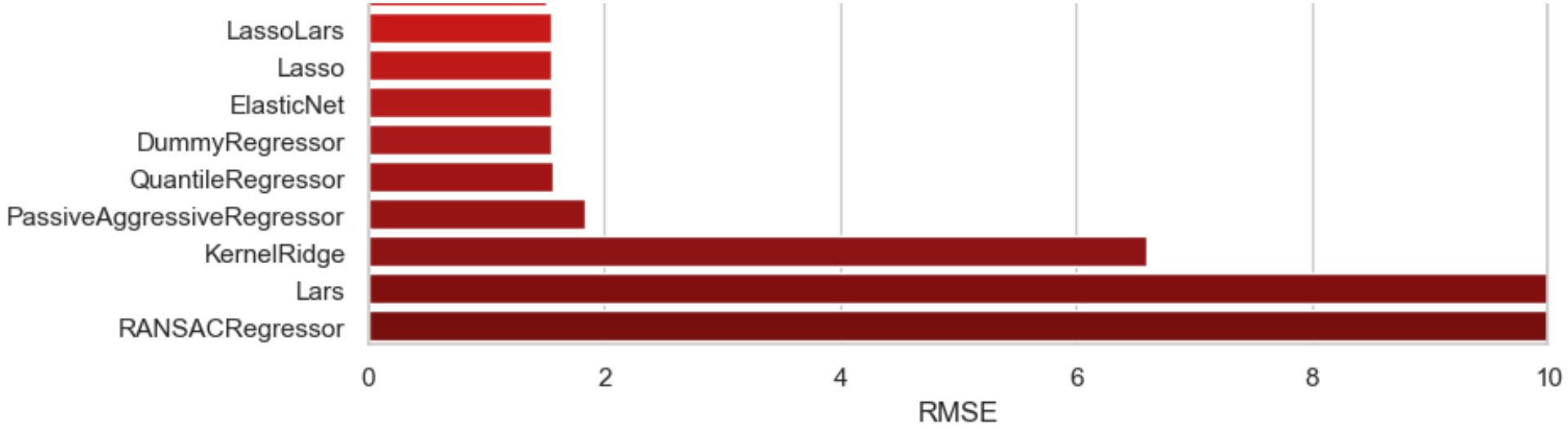
In [645…

```
plt.figure(figsize=(9, 12))
sns.set_theme(style="whitegrid")
ax = sns.barplot(y=predictions_train.index, x="RMSE", data=predictions_train,palette=sns.color_palette("seismic", len(predictions_t
ax.set(xlim=(0, 10))
```

Out[645…

```
[(0.0, 10.0)]
```

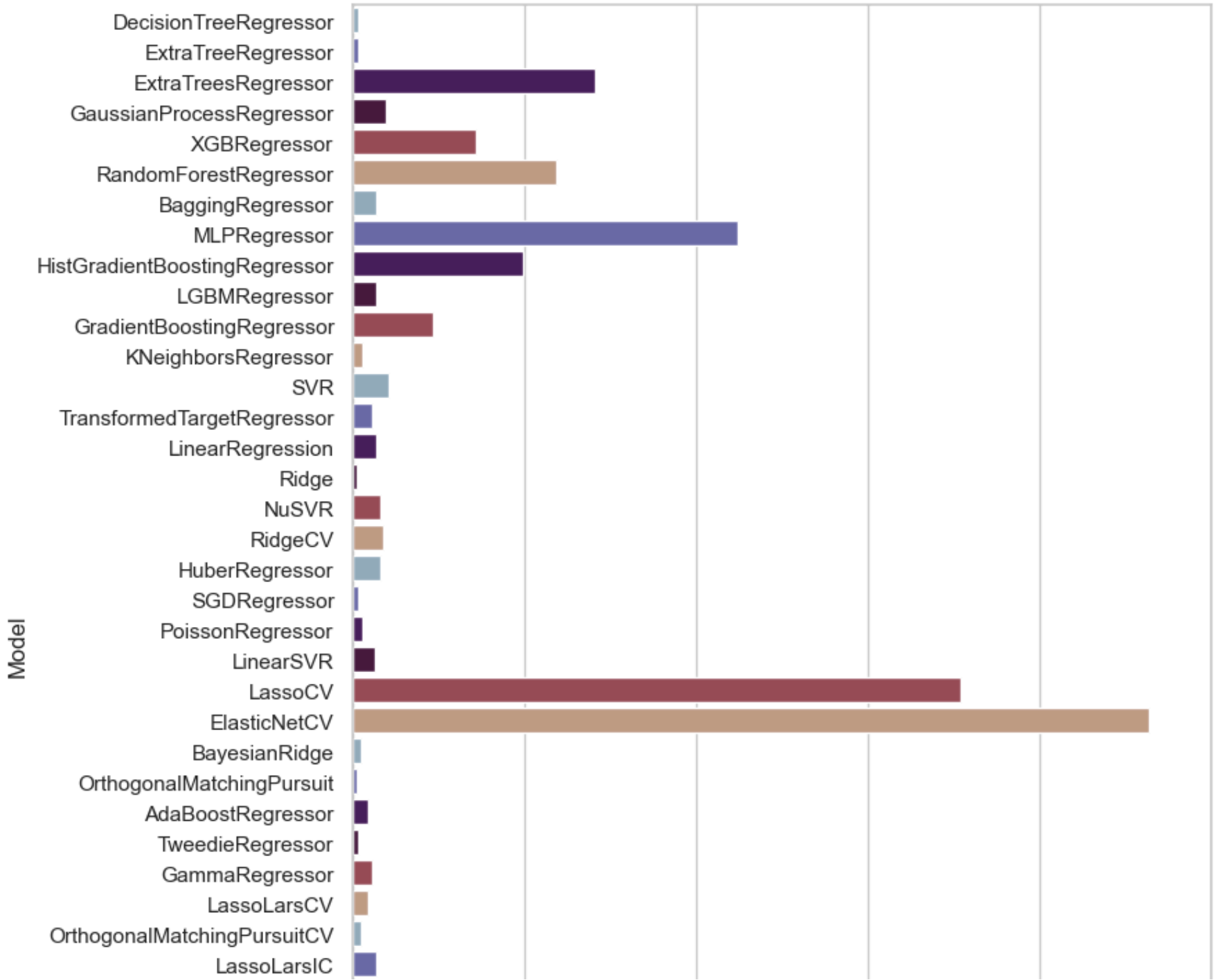

DecisionTreeRegressor
ExtraTreeRegressor
ExtraTreesRegressor
GaussianProcessRegressor
XGBRegressor
RandomForestRegressor
BaggingRegressor
MLPRegressor
HistGradientBoostingRegressor
LGBMRegressor
GradientBoostingRegressor
KNeighborsRegressor
SVR
TransformedTargetRegressor
LinearRegression
Ridge
NuSVR
RidgeCV
HuberRegressor
SGDRegressor
PoissonRegressor
LinearSVR
LassoCV
ElasticNetCV
BayesianRidge
OrthogonalMatchingPursuit
AdaBoostRegressor
TweedieRegressor
GammaRegressor
LassoLarsCV
OrthogonalMatchingPursuitCV
LassoLarsIC
LarsCV
Model

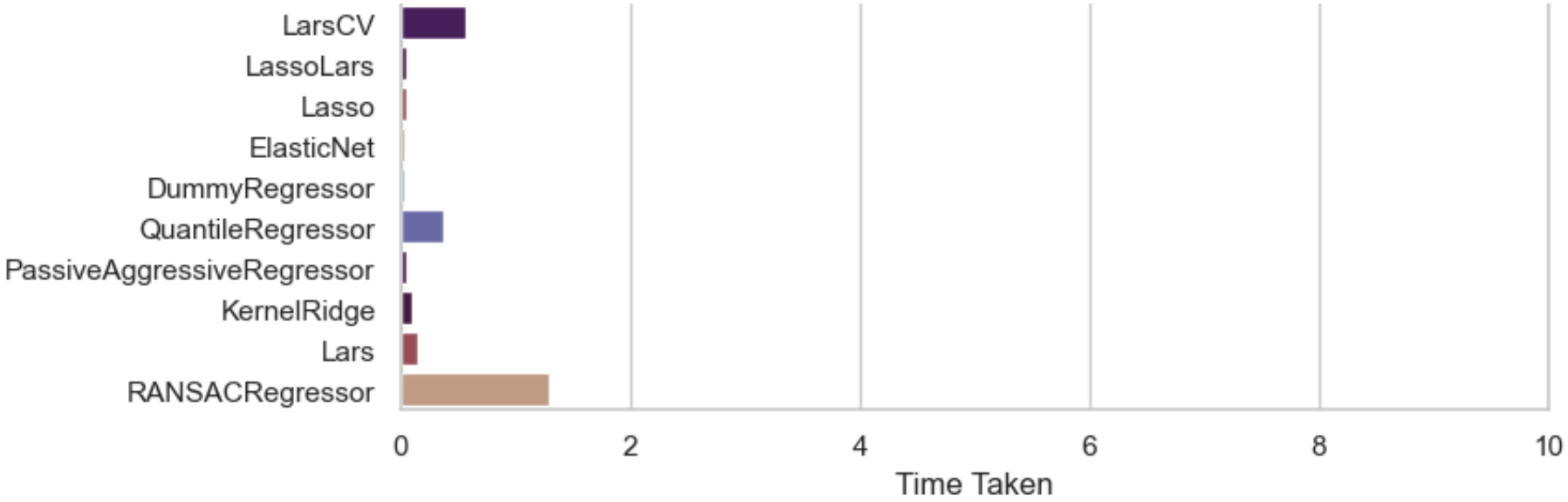


Bar plot of calculation time

In [649…

```
plt.figure(figsize=(8, 12))
sns.set_theme(style="whitegrid")

ax = sns.barplot(y=predictions_train.index, x="Time Taken", data=predictions_train,
                 palette=sns.color_palette("twilight"))

ax.set(xlim=(0, 10))
plt.show()
```


DecisionTreeRegressor
ExtraTreeRegressor
ExtraTreesRegressor
GaussianProcessRegressor
XGBRegressor
RandomForestRegressor
BaggingRegressor
MLPRegressor
HistGradientBoostingRegressor
LGBMRegressor
GradientBoostingRegressor
KNeighborsRegressor
SVR
TransformedTargetRegressor
LinearRegression
Ridge
NuSVR
RidgeCV
HuberRegressor
SGDRegressor
PoissonRegressor
LinearSVR
LassoCV
ElasticNetCV
BayesianRidge
OrthogonalMatchingPursuit
AdaBoostRegressor
TweedieRegressor
GammaRegressor
LassoLarsCV
OrthogonalMatchingPursuitCV
LassoLarsIC
Model


LarsCV
LassoLars
Lasso
ElasticNet
DummyRegressor
QuantileRegressor
PassiveAggressiveRegressor
KernelRidge
Lars
RANSACRegressor
0
2
4
6
8
10
Time Taken

In [ ]: