Model Comparison

```
In [ ]: import pymc3 as pm
```

```
In [1]: import pymc3 as pm
        import numpy as np
        import scipy.stats as stats
        import matplotlib.pyplot as plt
        import arviz as az
```

I am using pymc3

```
...\envs\pm3env\lib\site-packages\scipy\__init__.py:146: UserWa
...=1.16.5 and <1.23.0 is required for this version of SciPy (d
etected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
WARNING (theano.tensor.blas): Using NumPy C-API based implementation for BLAS functi
ons.
```

```
In [2]: az.style.use('arviz-darkgrid')
```

```
In [1]: import arviz as az
        import matplotlib.pyplot as plt
        import numpy as np
        import pymc as pm

        print(f"Running on PyMC v{pm.__version__}")
```

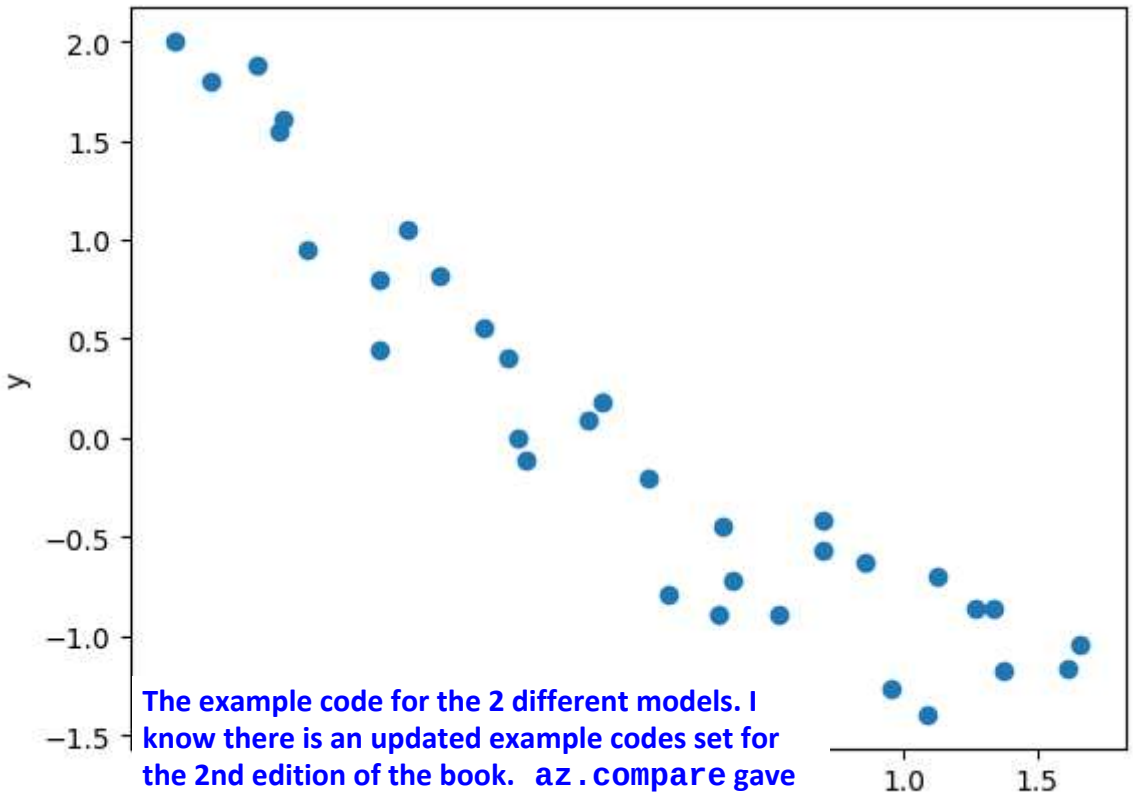
```
Running on PyMC v5.3.0
```

```
In [5]: x_1p
```

I also created a new conda environment and installed pymc v5 to see if I can get az.compare to work. It seems like I need to modify the example codes for the models as well. I haven't tried yet.

Out[5]:
```
array([[-1.0810000e+00, -8.8800000e-01, -6.2300000e-01, -4.8000000e-01,
        -5.0500000e-01, -3.4500000e-01,  5.2000000e-02,  6.1000000e-02,
         2.1500000e-01,  3.9500000e-01,  6.4300000e-01,  8.7100000e-01,
         7.7600000e-01,  8.3300000e-01,  1.2240000e+00,  1.3000000e+00,
         1.5630000e+00,  1.6690000e+00,  1.9670000e+00,  1.9500000e+00,
         2.0300000e+00,  2.2840000e+00,  2.5340000e+00,  2.5310000e+00,
         2.7650000e+00,  2.9130000e+00,  3.1680000e+00,  3.1190000e+00,
         3.3850000e+00,  3.4880000e+00,  3.5380000e+00,  3.9000000e+00,
         3.9700000e+00],
       [ 1.1685610e+00,  7.8854400e-01,  3.8812900e-01,  2.3040000e-01,
         2.5502500e-01,  1.1902500e-01,  2.7040000e-03,  3.7210000e-03,
         4.6225000e-02,  1.5602500e-01,  4.1344900e-01,  7.5864100e-01,
         6.0217600e-01,  6.9388900e-01,  1.4981760e+00,  1.6900000e+00,
         2.4429690e+00,  2.7855610e+00,  3.8690890e+00,  3.8025000e+00,
         4.1209000e+00,  5.2166560e+00,  6.4211560e+00,  6.4059610e+00,
         7.6452250e+00,  8.4855690e+00,  1.0036224e+01,  9.7281610e+00,
         1.1458225e+01,  1.2166144e+01,  1.2517444e+01,  1.5210000e+01,
         1.5760900e+01]])
```

In [2]:
```
dummy_data = np.loadtxt('../data/dummy.csv')
x_1 = dummy_data[:, 0]
y_1 = dummy_data[:, 1]

order = 2

x_1p = np.vstack([x_1**i for i in range(1, order+1)])
x_1s = (x_1p - x_1p.mean(axis=1, keepdims=True)) / \
    x_1p.std(axis=1, keepdims=True)
y_1s = (y_1 - y_1.mean()) / y_1.std()
plt.scatter(x_1s[0], y_1s)
plt.xlabel('x')
plt.ylabel('y')
plt.savefig('B11197_05_01.png', dpi=300)
```



The example code for the 2 different models. I know there is an updated example codes set for the 2nd edition of the book. `az.compare` gave me the same error there.

In [3]:
```
with pm.Model() as model_1:
    α = pm.Normal('α', mu=0, sd=1)
    β = pm.Normal('β', mu=0, sd=10)
    ϵ = pm.HalfNormal('ϵ', 5)

    μ = α + β * x_1s[0]

    y_pred = pm.Normal('y_pred', mu=μ, sd=ϵ, observed=y_1s)

    trace_1 = pm.sample(2000, return_inferencedata=True)

with pm.Model() as model_p:
    α = pm.Normal('α', mu=0, sd=1)
    β = pm.Normal('β', mu=0, sd=10, shape=order)
    ϵ = pm.HalfNormal('ϵ', 5)

    μ = α + pm.math.dot(β, x_1s)

    y_pred = pm.Normal('y_pred', mu=μ, sd=ϵ, observed=y_1s)

    trace_p = pm.sample(2000, return_inferencedata=True)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (4 chains in 4 jobs)
NUTS: [ϵ, β, α]
```

100.00% [12000/12000 00:14<00:00 Sampling 4 chains, 0 divergences]

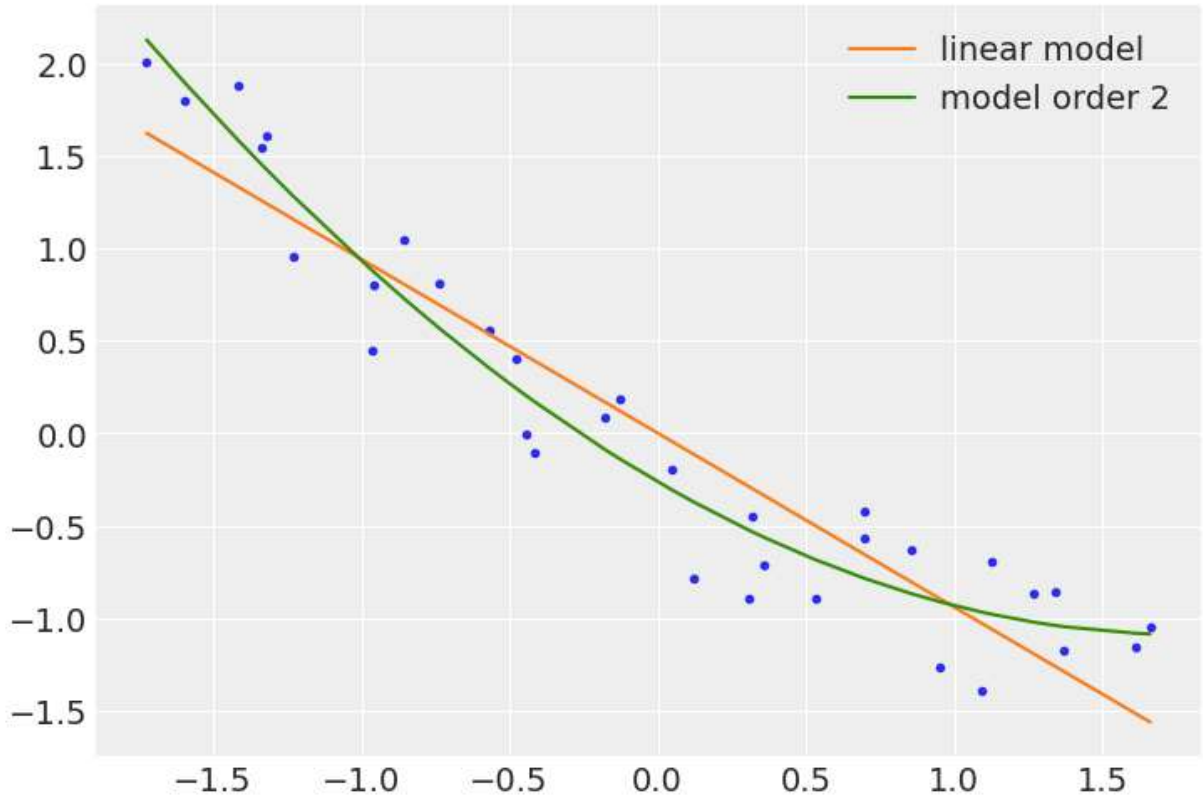
```
Sampling 4 chains for 1_000 tune and 2_000 draw iterations (4_000 + 8_000 draws tota
l) took 42 seconds.
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (4 chains in 4 jobs)
NUTS: [ϵ, β, α]
```

100.00% [12000/12000 00:10<00:00 Sampling 4 chains, 0 divergences]

```
Sampling 4 chains for 1_000 tune and 2_000 draw iterations (4_000 + 8_000 draws tota
l) took 29 seconds.
```

In [7]:

```
x_new = np.linspace(x_1s[0].min(), x_1s[0].max(), 100)

α_l_post = trace_l['α'].mean()
β_l_post = trace_l['β'].mean(axis=0)
y_l_post = α_l_post + β_l_post *  x_new

plt.plot(x_new, y_l_post, 'C1', label='linear model')

α_p_post = trace_p['α'].mean()
β_p_post = trace_p['β'].mean(axis=0)
idx = np.argsort(x_1s[0])
y_p_post = α_p_post + np.dot(β_p_post, x_1s)

plt.plot(x_1s[0][idx], y_p_post[idx], 'C2', label=f'model order {order}')

#α_p_post = trace_p['α'].mean()
#β_p_post = trace_p['β'].mean(axis=0)
#x_new_p = np.vstack([x_new**i for i in range(1, order+1)])
#y_p_post = α_p_post + np.dot(β_p_post, x_new_p)

plt.scatter(x_1s[0], y_1s, c='C0', marker='.')
plt.legend()
plt.savefig('B11197_05_02.png', dpi=300)
```



Posterior predictive checks

In [28]:
```
y_l = pm.sample_posterior_predictive(trace_l, 2000,
                                     model=model_l)['y_pred']

y_p = pm.sample_posterior_predictive(trace_p, 2000,
                                     model=model_p)['y_pred']
```

```
C:\Users\Mengj\anaconda3\envs\pm3env\lib\site-packages\pymc3\sampling.py:1708: UserW
arning: samples parameter is smaller than nchains times ndraws, some draws and/or ch
ains may not be represented in the returned posterior predictive sample
  warnings.warn(
```

100.00% [2000/2000 00:05<00:00]

100.00% [2000/2000 00:06<00:00]

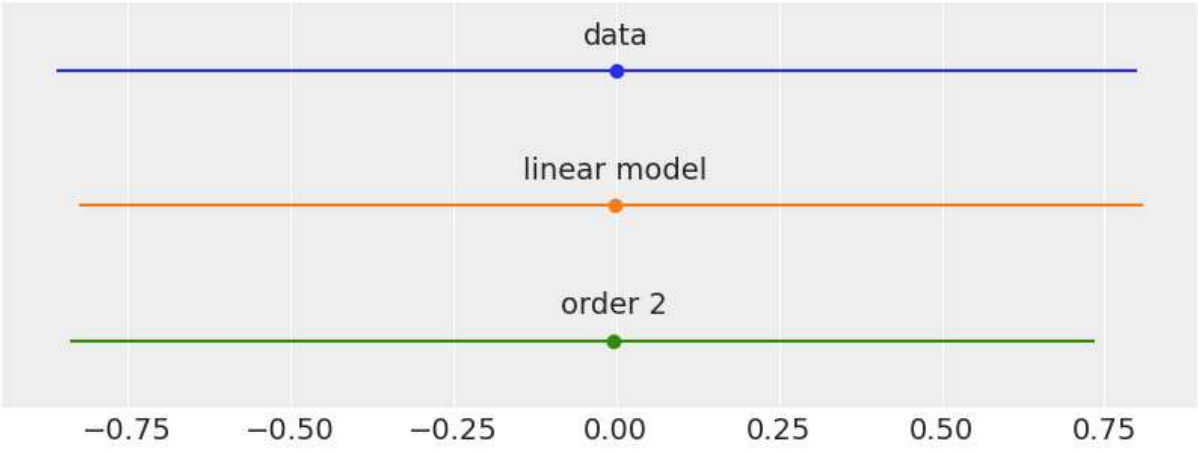
In [10]:
```
y_l.shape
```

Out[10]:
```
(2000, 33)
```

In [ ]:
```
for a, b in enumerate(data):
    print(a, b)
```

In [11]:
```
plt.figure(figsize=(8, 3))
data = [y_1s, y_l, y_p]
labels = ['data', 'linear model', 'order 2']
for i, d in enumerate(data):
    mean = d.mean()
```

```
    err = np.percentile(d, [25, 75])
    plt.errorbar(mean, -i, xerr=[[-err[0]], [err[1]]], fmt='o')
    plt.text(mean, -i+0.2, labels[i], ha='center', fontsize=14)
plt.ylim([-i-0.5, 0.5])
plt.yticks([])
plt.savefig('B11197_05_03.png', dpi=300)
```



In [17]:
```
np.percentile(y_1s, [75, 25])
```

Out[17]:
```
array([ 0.79986227, -0.85902444])
```

In [18]:
```
0.79986227 + 0.85902444
```

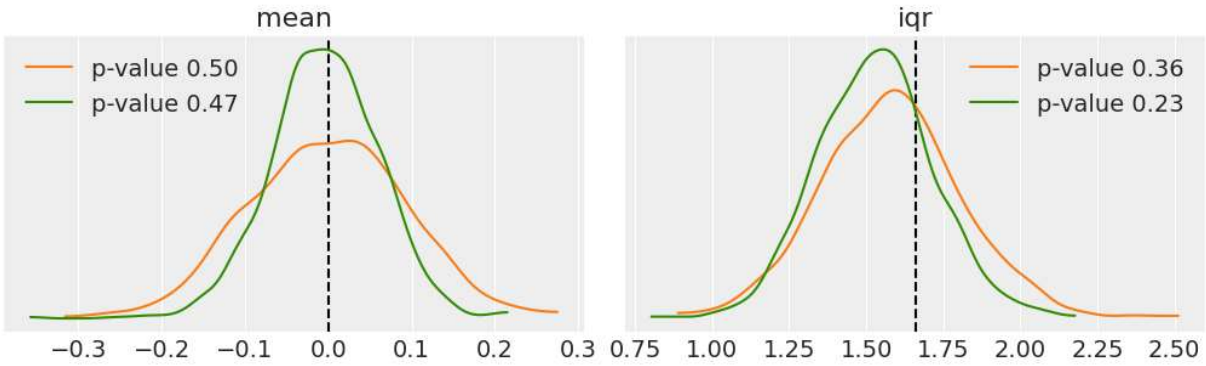
Out[18]:
```
1.65888671
```

In [23]:
```
for idx, func in enumerate([np.mean, iqr]):
    T_obs = func(y_1s)
    print(T_obs)
    for d_sim, c in zip([y_l, y_p], ['C1', 'C2']):
        T_sim = func(d_sim, 1)
        print(T_sim)
```

```
-5.3828995133340925e-17
[-0.12296482  0.10256937 -0.01599578 ... -0.06025549  0.02738558
  0.12141346]
[-0.09601874 -0.00740317  0.06423412 ... -0.10349315  0.01971963
 -0.03471059]
1.6588867027733327
[1.89574617 1.56502387 1.3809935  ... 1.57936058 1.87361482 1.8443778 ]
[1.64032166 1.6707949  1.55701969 ... 1.31546495 1.65016167 1.52054638]
```

In [13]:
```
fig, ax = plt.subplots(1, 2, figsize=(10, 3), constrained_layout=True)


def iqr(x, a=0):
    return np.subtract(*np.percentile(x, [75, 25], axis=a))


for idx, func in enumerate([np.mean, iqr]):
    T_obs = func(y_1s)
    ax[idx].axvline(T_obs, 0, 1, color='k', ls='--')
```

```
    for d_sim, c in zip([y_l, y_p], ['C1', 'C2']):
        T_sim = func(d_sim, 1)
        p_value = np.mean(T_sim >= T_obs)
        az.plot_kde(T_sim, plot_kwargs={'color': c},
                    label=f'p-value {p_value:.2f}', ax=ax[idx])
    ax[idx].set_title(func.__name__)
    ax[idx].set_yticks([])
    ax[idx].legend()
plt.savefig('B11197_05_04.png', dpi=300)
```

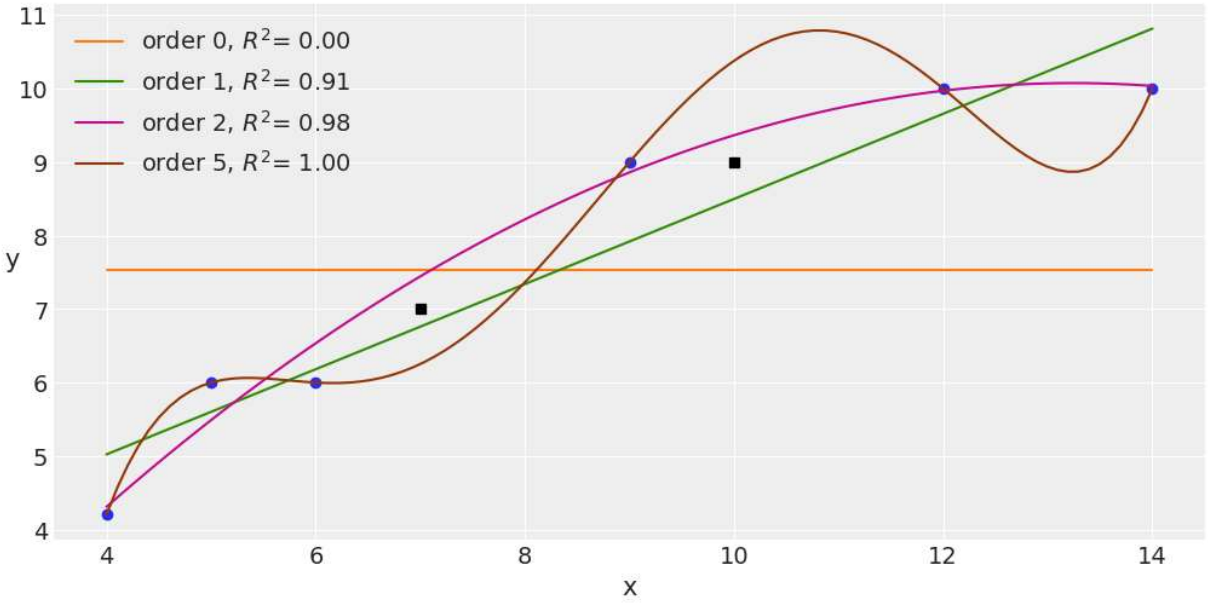


Occam's razor – simplicity and accuracy

In [24]:

```
x = np.array([4., 5., 6., 9., 12, 14.])
y = np.array([4.2, 6., 6., 9., 10, 10.])

plt.figure(figsize=(10, 5))
order = [0, 1, 2, 5]
plt.plot(x, y, 'o')
for i in order:
    x_n = np.linspace(x.min(), x.max(), 100)
    coeffs = np.polyfit(x, y, deg=i)
    ffit = np.polyval(coeffs, x_n)

    p = np.poly1d(coeffs)
    print(p)
    yhat = p(x)
    ybar = np.mean(y)
    ssreg = np.sum((yhat-ybar)**2)
    sstot = np.sum((y - ybar)**2)
    r2 = ssreg / sstot

    plt.plot(x_n, ffit, label=f'order {i}, $R^2$= {r2:.2f}')

plt.legend(loc=2)
plt.xlabel('x')
plt.ylabel('y', rotation=0)
plt.savefig('B11197_05_05.png', dpi=300)
plt.plot([10, 7], [9, 7], 'ks')
plt.savefig('B11197_05_06.png', dpi=300)
```

```
7.533

0.5795 x + 2.704
          2
-0.06751 x + 1.788 x - 1.764
          5          4         3          2
0.004155 x - 0.1848 x + 3.14 x - 25.39 x + 98.26 x - 140.5
```



Computing information criteria with PyMC3

In [4]:
```
waic_l = az.waic(trace_l)
waic_l
```

Out[4]:
```
Computed from 8000 posterior samples and 33 observations log-likelihood matrix.

          Estimate       SE
elpd_waic   -14.35     2.68
p_waic        2.44        -
```

In [37]:
```
trace_l
```

Out[37]: arviz.InferenceData

- ► posterior
- ► log_likelihood
- ► sample_stats
- ► observed_data

In [5]:
```
cmp_df = az.compare({'model_l':trace_l, 'model_p':trace_p},method='BB-pseudo-BMA',
cmp_df
```

The error message I got from `az.compare` is in the following pages. The whole line of code wasn't printed out completely here, though...

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\core\indexes\base.py:3652, in
Index.get_loc(self, key)
   3651 try:
-> 3652     return self._engine.get_loc(casted_key)
   3653 except KeyError as err:

File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\_libs\index.pyx:147, in panda
s._libs.index.IndexEngine.get_loc()

File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\_libs\index.pyx:153, in panda
s._libs.index.IndexEngine.get_loc()

TypeError: 'slice(None, None, None)' is an invalid key

During handling of the above exception, another exception occurred:

InvalidIndexError                         Traceback (most recent call last)
File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\core\frame.py:4189, in DataFra
me._set_value(self, index, col, value, takeable)
   4188 else:
-> 4189     icol = self.columns.get_loc(col)
   4190     iindex = self.index.get_loc(index)

File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\core\indexes\base.py:3659, in
Index.get_loc(self, key)
   3655 except TypeError:
   3656     # If we have a listlike key, _check_indexing_error will raise
   3657     #  InvalidIndexError. Otherwise we fall through and re-raise
   3658     #  the TypeError.
-> 3659     self._check_indexing_error(key)
   3660     raise

File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\core\indexes\base.py:5736, in
Index._check_indexing_error(self, key)
   5733 if not is_scalar(key):
   5734     # if key is not a scalar, directly raise an error (the code below
   5735     # would convert to numpy arrays and raise later any way) - GH29926
-> 5736     raise InvalidIndexError(key)

InvalidIndexError: slice(None, None, None)

The above exception was the direct cause of the following exception:

InvalidIndexError                         Traceback (most recent call last)
Cell In[5], line 1
----> 1 cmp_df = az.compare({'model_1':trace_1, 'model_p':trace_p},method='BB-pseudo
-BMA', ic="waic", scale="deviance")
      2 cmp_df

File ~\anaconda3\envs\pm3env\lib\site-packages\arviz\stats\stats.py:306, in compare
(compare_dict, ic, method, b_samples, alpha, seed, scale, var_name)
    304         std_err = ses.loc[val]
    305         weight = weights[idx]
--> 306         df_comp.at[val] = (
```

```
    307             idx,
    308             res[ic],
    309             res[p_ic],
    310             d_ic,
    311             weight,
    312             std_err,
    313             d_std_err,
    314             res["warning"],
    315             res[scale_col],
    316         )
    318 df_comp["rank"] = df_comp["rank"].astype(int)
    319 df_comp["warning"] = df_comp["warning"].astype(bool)

File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\core\indexing.py:2423, in _AtIndexer.__setitem__(self, key, value)
   2420     self.obj.loc[key] = value
   2421     return
-> 2423 return super().__setitem__(key, value)

File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\core\indexing.py:2379, in _ScalarAccessIndexer.__setitem__(self, key, value)
   2376 if len(key) != self.ndim:
   2377     raise ValueError("Not enough indexers for scalar access (setting)!")
-> 2379 self.obj._set_value(*key, value=value, takeable=self._takeable)

File ~\anaconda3\envs\pm3env\lib\site-packages\pandas\core\frame.py:4209, in DataFrame._set_value(self, index, col, value, takeable)
   4204     self._item_cache.pop(col, None)
   4206 except InvalidIndexError as ii_err:
   4207     # GH48729: Seems like you are trying to assign a value to a
   4208     # row when only scalar options are permitted
-> 4209     raise InvalidIndexError(
   4210         f"You can only assign a scalar value not a {type(value)}"
   4211     ) from ii_err

InvalidIndexError: You can only assign a scalar value not a <class 'tuple'>
```