PSY9511: Seminar 4

Model selection, validation and testing

Esten H. Leonardsen

23.09.24

UNIVERSITETET I OSLO

Strategies for model evaluation

UNIVERSITETET
I OSLO

Model evaluation: Rationale

Statistical inference:

Goal: In-sample quantification

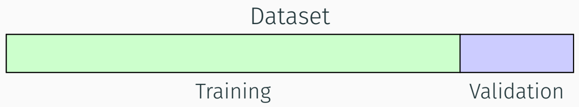
Predictive modelling:

Goal: Out-of-sample generalization

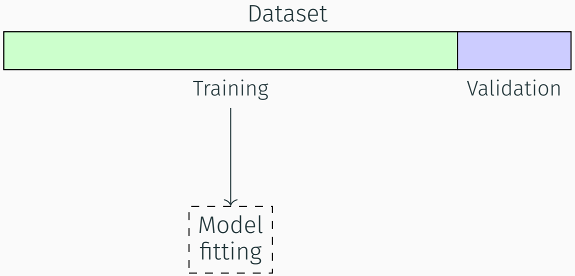
- How can we test how good our model is on **unseen data** and **be certain that performance holds if we present even more new data**

Model evaluation: Validation set

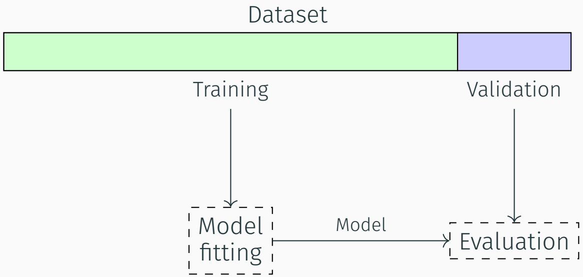
Model evaluation: Validation set



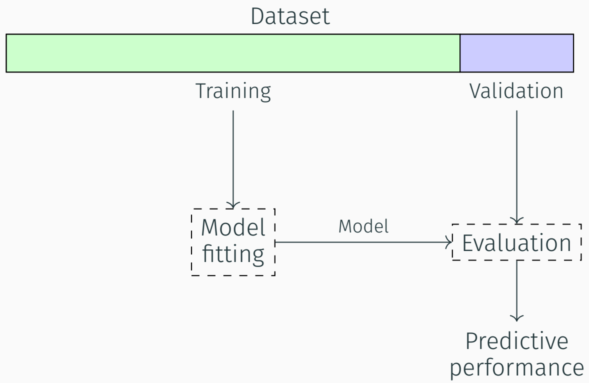
Model evaluation: Validation set



Model evaluation: Validation set



Model evaluation: Validation set

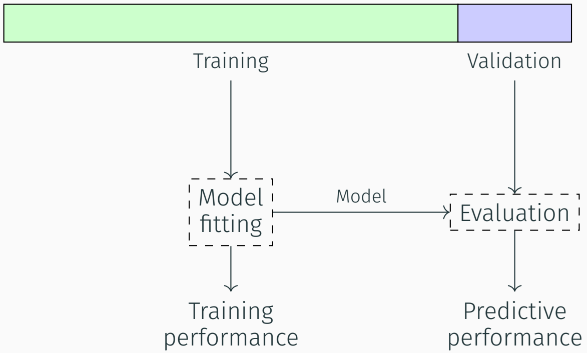


Model evaluation: Validation set

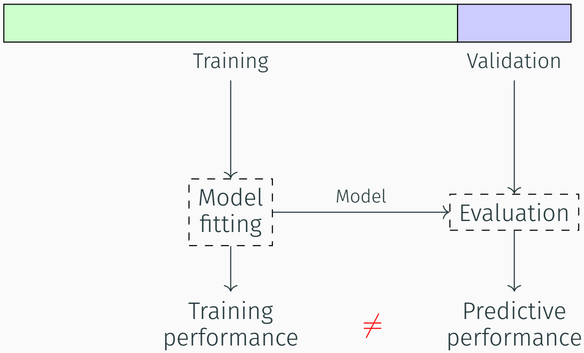
In the validation set approach we split the dataset into two subsets (commonly ∼80%/20%), use the first for training the model and the second to test its performance.

+ Accurate estimate of out-of-sample error
+ Simple
- Variable results depending on the exact split
- Only uses a subset of data for training models
- Gives a point estimate of the error, without confidence intervals

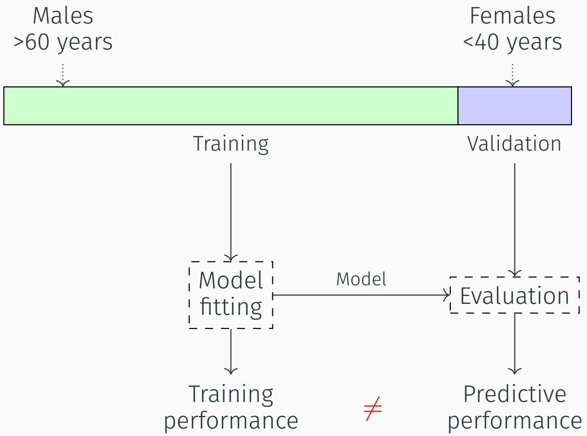
Model evaluation: Validation set



Model evaluation: Validation set



Model evaluation: Validation set



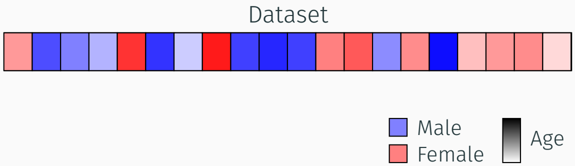
Model evaluation: Stratification

Stratification:
Ensuring all folds of the dataset are similar with respect to some given characteristics.

Model evaluation: Stratification

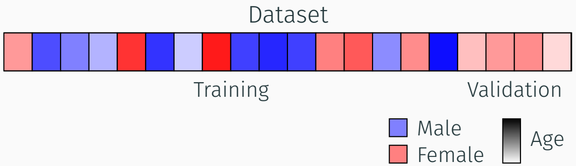
In[1]:
```
df = ...
```

Model evaluation: Stratification



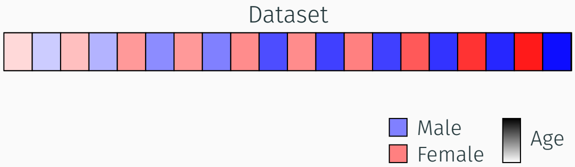
In[1]:
```
df = ...
```

Model evaluation: Stratification



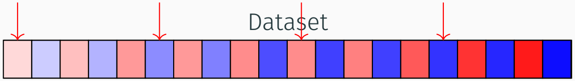
In[1]:
```
df = ...

train = df.iloc[:int(len(df) * 0.8)]
validation = df.iloc[int(len(df) * 0.8):]
```

Model evaluation: Stratification



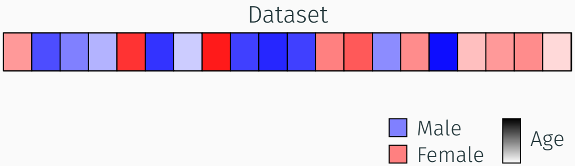
In[1]:
```
df = ...
df = df.sort_values(['sex', 'age'])
```

Model evaluation: Stratification



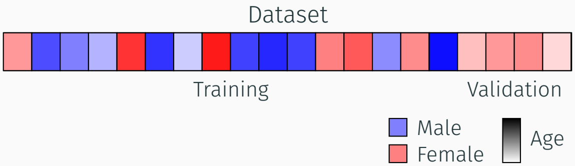
In[1]:
```
df = ...
df = df.sort_values(['sex', 'age'])

df['fold'] = np.arange(len(df)) % (1 / 0.2)
train = df[df['fold'] != 0]
val = df[df['fold'] == 0]
```

Model evaluation: Stratification



In[1]:
```
df = ...
```

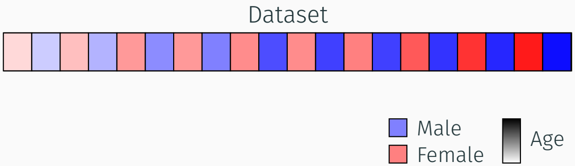
Model evaluation: Stratification



In[1]:

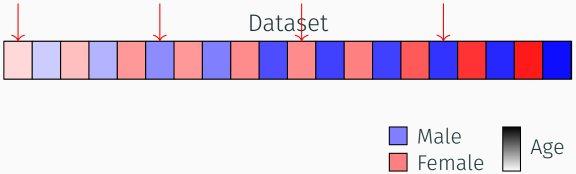
```
df = ...

train = df.iloc[:int(len(df) * 0.8)]
validation = df.iloc[int(len(df) * 0.8):]
```

Model evaluation: Stratification



In[1]:
```
df = ...
df = df.sort_values(['sex', 'age'])
```

Model evaluation: Stratification



In[1]:
```
df = ...
df = df.sort_values(['sex', 'age'])

df['fold'] = np.arange(len(df)) % (1 / 0.2)
train = df[df['fold'] != 0]
val = df[df['fold'] == 0]
```

Model evaluation: Stratification

Stratification:

Ensuring all folds of the dataset are similar with respect to some given characteristics.

- Helps alleviate the risk of training performance $\gg$ validation performance
- **Always** stratify on target variable first
- Also good idea to stratify on other core characteristics, e.g. sex and age

```
In[1]:  from sklearn.model_selection import  train_test_split
```

```
library(splitstackshape)
stratified(data, columns, split)
```