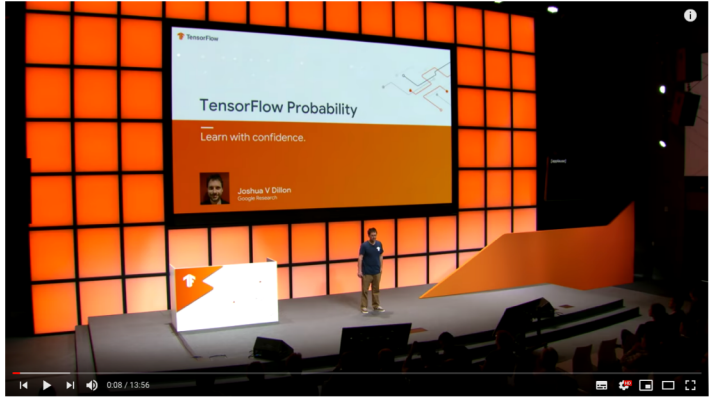
Zurich University
of Applied Sciences

Uncertainty in DL models
Latest Development in TF Probability

Oliver Dürr

Remark: Much of the material has been developed together with Elvis Murina and Beate Sick

Topics



[TF Probability Talk 6/3/2019]

- Sneak preview into TF Probability
- Demo:
  - Linear Regression (I) The usual probabilistic way
  - Linear Regression (II) Non-Fixed Variance
  - Linear Regression (III) Bayesian Way

Disclaimer: Fresh from the loom, might be a bit rough

Docker:

```
docker pull oduerr/tf_docker:cpu_r_tfp
```

Recap: Linear regression in statistics

Model for the conditional probability distribution

CPD: $Y_{X_i} = (Y|X_i) \sim N(\mu_{x_i}, \sigma^2)$

$Y_x \in \sim$

$\mu_x \in \sim$

μ_x is modeled

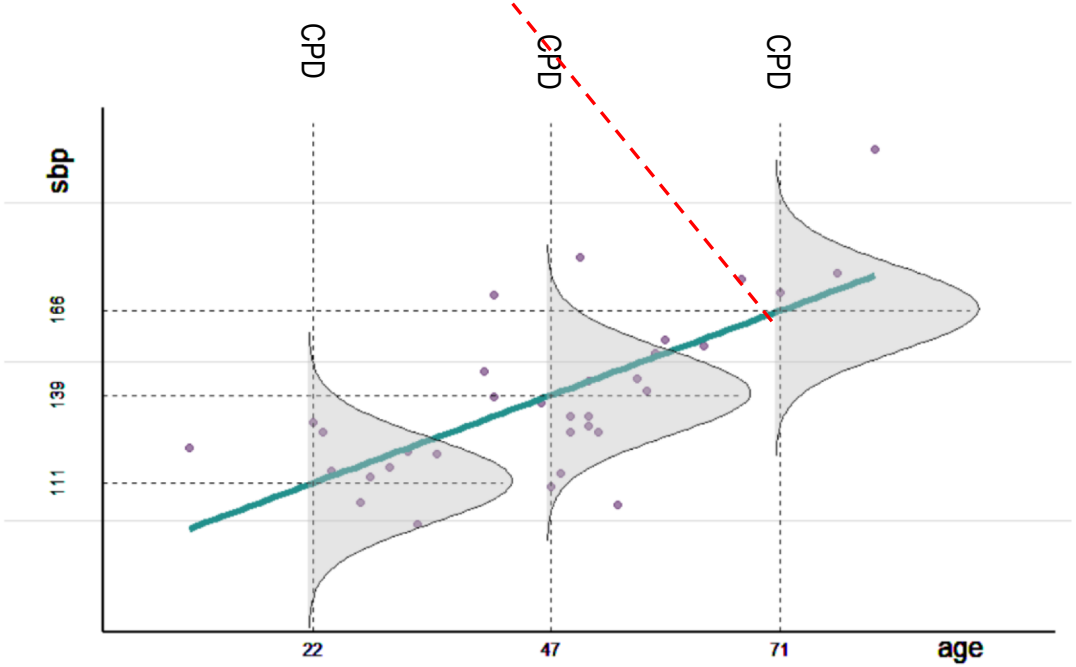
σ^2 is independent of the predictor values

$$y_i = \beta_0 + \beta_1 \cdot x_{i1} + \varepsilon_i$$

$$E(Y_{X_i}) = \mu_{x_i} = (\mu|X=x_i) = \beta_0 + \beta_1 \cdot x_{i1}$$

$$Var(Y_{X_i}) = Var(Y|X_i) = Var(\varepsilon_i) = \sigma^2$$

$$\varepsilon_i \text{ i.i.d.} \sim N(0, \sigma^2)$$



Demo also sigma

TFP: Neural Network Parametrizes Distributiuon

Model for the conditional probability distribution

CPD: $Y_{X_i} = (Y|X_i) \sim N(\mu_{x_i}, \sigma^2)$

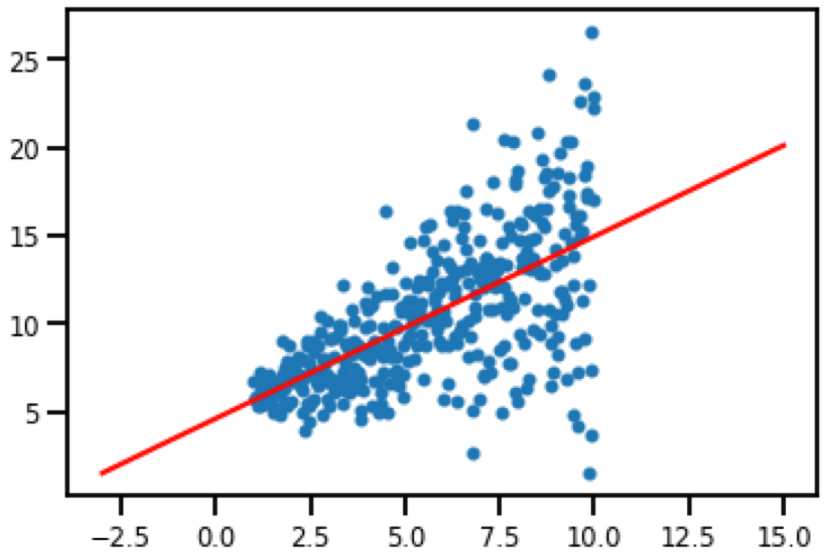
Sigma is fixed

```
In [201]: def my_gauss(mu):
            return tfd.Normal(loc=mu, scale=1)

          #Using the functional API for clarity
          inputs = tf.keras.layers.Input(shape=(1,))
          mu = tf.keras.layers.Dense(1)(inputs)
          p_y = tfp.layers.DistributionLambda(my_gauss)(mu)

          model = tf.keras.models.Model(inputs=inputs, outputs=p_y)
```

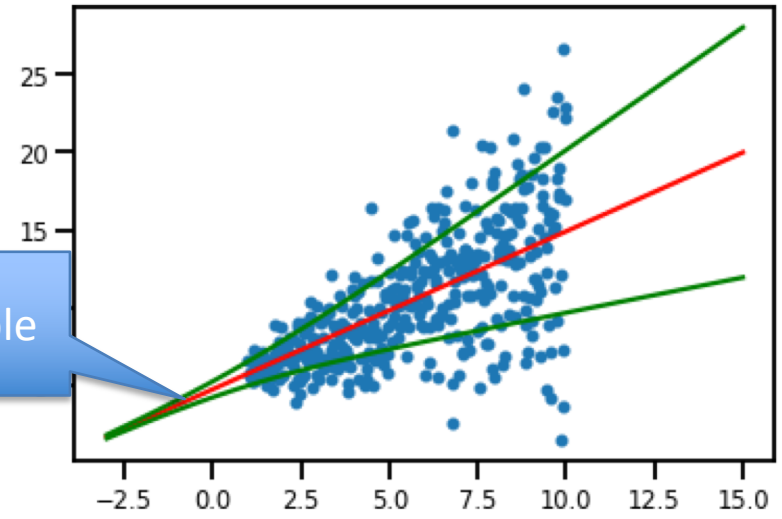
Also modeling sigma

```
def my_gauss2(t):
  mu = t[...,:1] #First column mu
  scale = 1e-3 + tf.math.softplus(0.05 * t[...,1:]) #Second column sigma
  return tfd.Normal(loc=mu, scale=scale)

#Using the functional API for clarity
inputs = tf.keras.layers.Input(shape=(1,))
t = tf.keras.layers.Dense(2)(inputs) #<-- 2 outputs
p_y = tfp.layers.DistributionLambda(my_gauss2)(t)
model = tf.keras.models.Model(inputs=inputs, outputs=p_y)
```

Results



No modelling of sigma

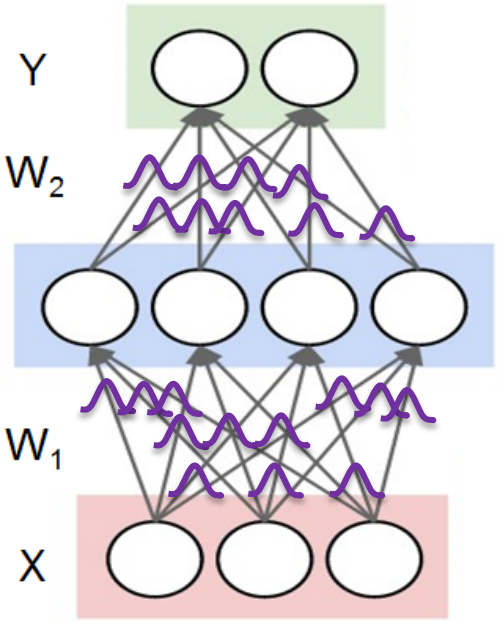


Modelling of sigma

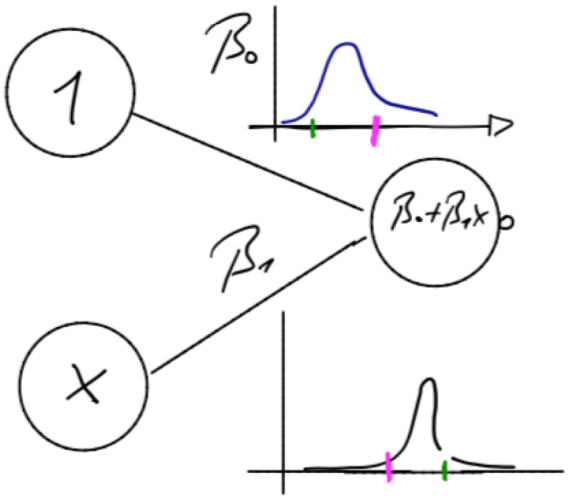
Questionable

Bayesian DL

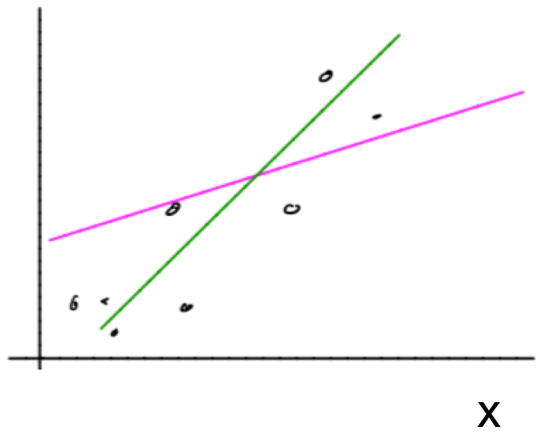
Modelling Uncertainty: Linear Bayes



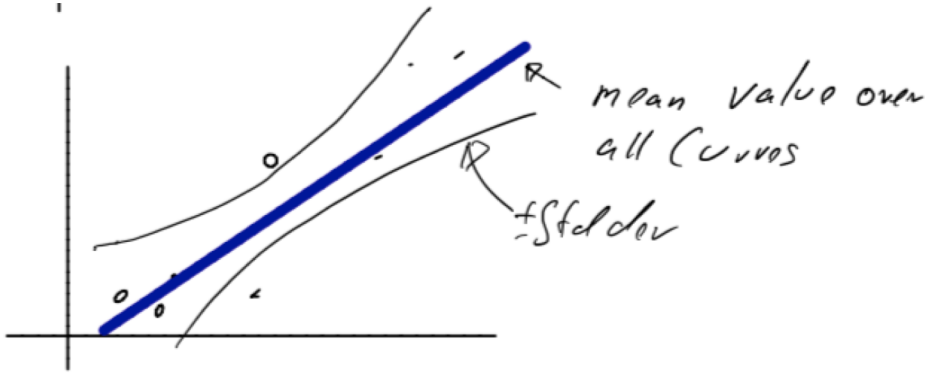
Bayesian NN



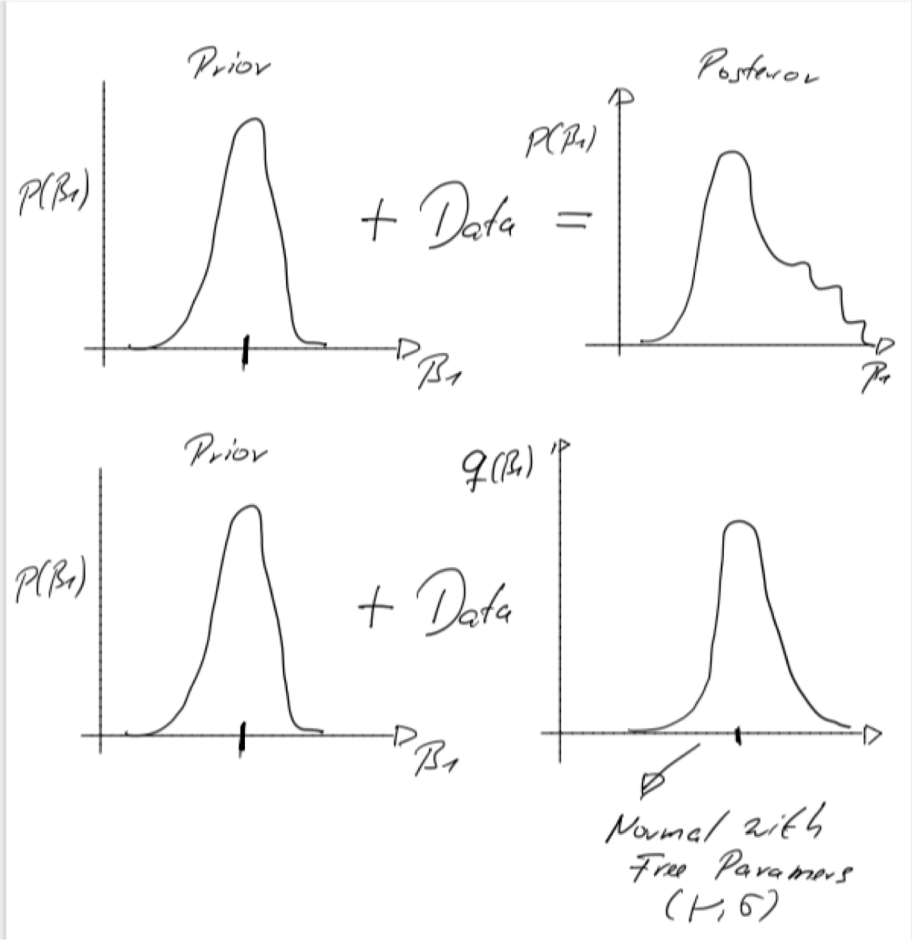
2 Examples



Averages over all



Training Using the variational approximation



"Normal" Bayes

"Variational Approximation" Bayes

R code for Variational Bayes

Before: no

```
In [201]: def my_gauss(mu):
            return tfd.Normal(loc=mu, scale=1)

          #Using the functional API for clarity
          inputs = tf.keras.layers.Input(shape=(1,))
          mu = tf.keras.layers.Dense(1)(inputs)
          p_y = tfp.layers.DistributionLambda(my_gauss)(mu)

          model = tf.keras.models.Model(inputs=inputs, outputs=p_y)
```

```
#Using the functional API for clarity
inputs = tf.keras.layers.Input(shape=(1,))
tt = tfp.layers.DenseVariational(1, posterior_mean_field, prior_trainable)(inputs)
p_y = tfp.layers.DistributionLambda(my_gauss)(tt)

model = tf.keras.models.Model(inputs=inputs, outputs=p_y)
```