

Prediction of Electrical Power Output: A Bayesian Statistics Approach

Introduction

Predicting electrical power output of a combined cycle power plant (CCPP) is important in order to maximize the profit. A combined cycle power plant layout is shown in Figure 1 [1]. For accurate system analysis with thermo-dynamical approaches, a high number of assumptions is necessary and compels thousands of nonlinear equations, whose solution is either almost impossible or takes too much computational time and efforts.

To eliminate this barrier, this study utilizes Bayesian multiple regression model to exam covariates and develop a predictive model. The data file (Folds5x2_pp.xlsx) is downloaded from UCI machine learning repository [2], which consists of 9568 data points collected from a CCPP over 6 years (2006-2011). The electrical power output (EP) is sensitive to ambient temperature (AT), atmospheric pressure (AP), relative humidity (RH) and exhaust vacuum (V):

- Ambient Temperature (AT) in the range of 1.81°C and 37.11°C,
- Atmospheric Pressure (AP) in the range 992.89-1033.30 milibar,
- Relative Humidity (RH) in the range 25.56% to 100.16%,
- Exhaust Vacuum (V) in teh range 25.36-81.56 cm Hg,
- Net hourly electrical energy output (EP) 420-495.76 MW.

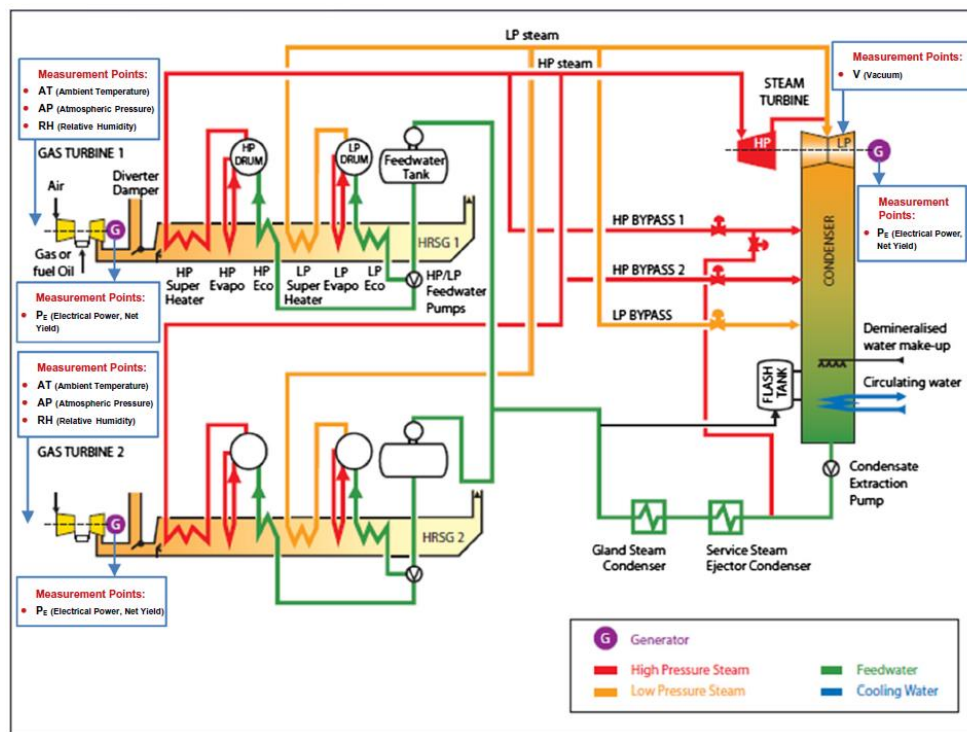


Figure 1: The Combined Cycle Power Plant Layout

Pre-processing Data

For illustration purposes, only the first 60 data sets are used in this study. Data pre-processing includes training-validating data split and outlier diagnostics:

1. Training-validating Data Split:

The data split is performed using Python code (ProjectPreprocessData.py) as shown in Figure A1. The “train_test_split” from sklearn.model_selection is used and randomly allocates 40 data sets to training data and the other 20 data sets to validating data. The processed data is copied into OpenBUGS data section.

2. Outlier Diagnostics using Conditional Predictive Ordinate (CPO):

The CPO of training data is analyzed using OpenBUGS code (ProjectTrainCPO.odc), see Figure A2. The CPO of 32 and 33, 0.008 (1/125.2) and 0.009 (1/113.9) in Figure 2, are less than 1%, so they are the outliers and removed from training data in the future analysis.

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
icpo[1]	28.76	9.107	0.03827	17.68	26.71	51.86	1001	1000000
icpo[2]	26.91	11.76	0.08788	15.15	23.78	57.23	1001	1000000
icpo[3]	24.85	11.7	0.08626	14.2	21.61	54.67	1001	1000000
icpo[4]	18.0	3.524	0.01827	13.19	17.34	26.77	1001	1000000
icpo[5]	33.02	24.56	0.1646	14.99	26.24	91.81	1001	1000000
icpo[6]	15.76	2.287	0.009297	12.21	15.46	21.08	1001	1000000
icpo[7]	27.56	16.78	0.1225	14.27	22.81	69.29	1001	1000000
icpo[8]	33.48	13.38	0.05243	18.47	30.2	67.88	1001	1000000
icpo[9]	23.32	5.177	0.02582	16.21	22.35	36.11	1001	1000000
icpo[10]	31.15	20.64	0.1523	14.96	25.41	81.59	1001	1000000
icpo[11]	15.77	2.226	0.01262	12.25	15.49	20.92	1001	1000000
icpo[12]	16.87	3.343	0.02078	12.52	16.21	25.25	1001	1000000
icpo[13]	18.07	5.705	0.04259	12.58	16.67	32.43	1001	1000000
icpo[14]	19.98	3.892	0.0216	14.49	19.29	29.51	1001	1000000
icpo[15]	16.98	2.638	0.0189	12.95	16.61	23.21	1001	1000000
icpo[16]	25.13	9.523	0.04599	15.01	22.69	49.66	1001	1000000
icpo[17]	19.26	4.696	0.01858	13.47	18.22	31.21	1001	1000000
icpo[18]	15.15	1.903	0.006966	11.97	14.96	19.42	1001	1000000
icpo[19]	53.74	33.17	0.2011	22.54	44.86	137.9	1001	1000000
icpo[20]	16.98	3.593	0.01321	12.51	16.25	26.03	1001	1000000
icpo[21]	16.18	2.326	0.01132	12.51	15.89	21.57	1001	1000000
icpo[22]	19.55	4.264	0.0269	13.88	18.71	30.21	1001	1000000
icpo[23]	65.12	88.27	0.4883	17.8	43.01	244.8	1001	1000000
icpo[24]	32.93	17.05	0.07588	16.6	28.38	76.26	1001	1000000
icpo[25]	26.07	11.21	0.09826	14.91	23.09	54.93	1001	1000000
icpo[26]	15.94	2.546	0.0133	12.21	15.55	22.02	1001	1000000
icpo[27]	15.55	2.145	0.01045	12.13	15.29	20.47	1001	1000000
icpo[28]	15.99	2.561	0.01122	12.25	15.6	22.07	1001	1000000
icpo[29]	83.0	54.81	0.3128	31.47	68.33	222.4	1001	1000000
icpo[30]	20.43	4.739	0.03979	14.26	19.45	32.37	1001	1000000
icpo[31]	46.69	25.62	0.1405	21.23	39.96	112.2	1001	1000000
icpo[32]	125.2	141.8	0.8371	30.87	87.31	447.6	1001	1000000
icpo[33]	113.9	104.8	0.3722	33.17	85.41	367.4	1001	1000000
icpo[34]	67.05	41.62	0.277	27.12	56.0	172.5	1001	1000000
icpo[35]	15.81	2.437	0.01193	12.16	15.45	21.55	1001	1000000
icpo[36]	22.28	5.137	0.01865	15.41	21.27	35.1	1001	1000000
icpo[37]	24.77	17.46	0.1135	13.3	19.85	65.69	1001	1000000
icpo[38]	18.68	4.391	0.01626	13.22	17.73	29.79	1001	1000000
icpo[39]	20.54	5.09	0.04157	14.09	19.44	33.41	1001	1000000
icpo[40]	15.49	2.075	0.008698	12.13	15.25	20.23	1001	1000000

Figure 2: CPO Results of Training Data

Bayesian Multiple Regression Models and Results

1. Model 1 with All Covariates:

The Bayesian multiple regression model and training data are used to develop the predictive model. Model 1 includes all covariates as following:

Data in the form: $(ATt_i, Vt_i, APt_i, RHt_i, EPt_i)$

Multiple regression model: $EPt_i = \alpha + \beta_1 \cdot ATt_i + \beta_2 \cdot Vt_i + \beta_3 \cdot APt_i + \beta_4 \cdot RHt_i + \varepsilon_i$
 $\varepsilon_i \sim N(0, \sigma^2)$

All parameters use non-informative priors:

$$\alpha \sim N(0, 0.001), \beta_i \sim N(0, 0.001) \text{ and } \tau \sim Ga(0.001, 0.001)$$

The simulation is performed in OpenBUGS (ProjectMultReg1.odc, see Figure A3) and the estimated α and β s are shown in Figure 3. The boxplot of β s from Model 1, see Figure 4, shows that the covariate of Exhaust Vacuum (V) is not significant.

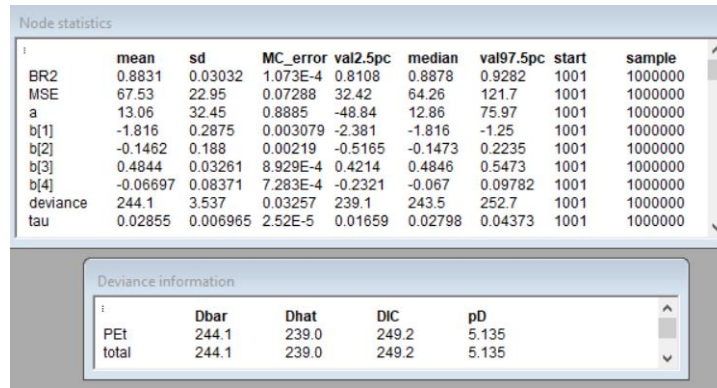


Figure 3: Simulated Results of Model 1

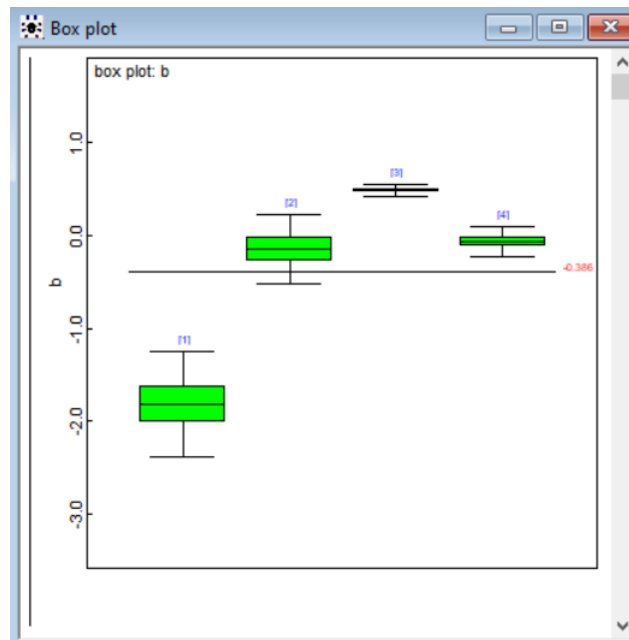


Figure 4: Boxplot of β s in Model 1

2. Model 2 w/o Covariate of Exhaust Vacuum (V):

Thus Model 2 excludes the covariate of Exhaust Vacuum (V):

$$EPt_i = \alpha + \beta_1 \cdot ATt_i + \beta_2 \cdot APt_i + \beta_3 \cdot RHT_i + \varepsilon_i \text{ and } \varepsilon_i \sim N(0, \sigma^2)$$

The simulation is performed in OpenBUGS (ProjectMultReg2.odc, see Figure A3) and the estimated α and β s are shown in Figure 5. The boxplot of β s from Model 2, see Figure 6, shows that all three covariates are significant.

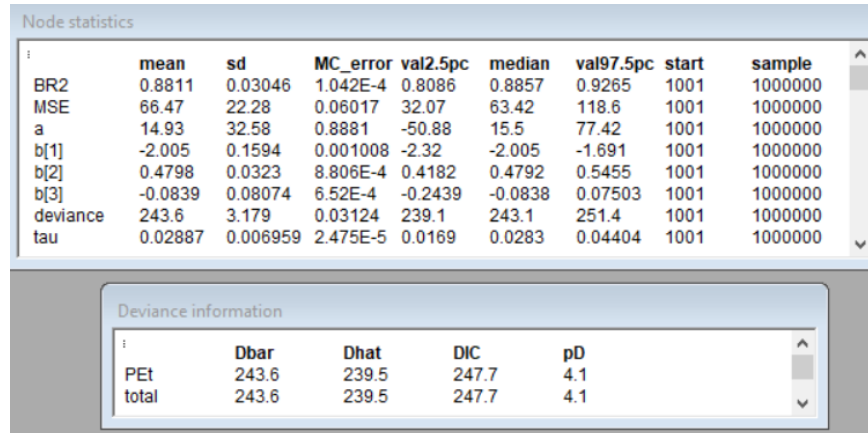
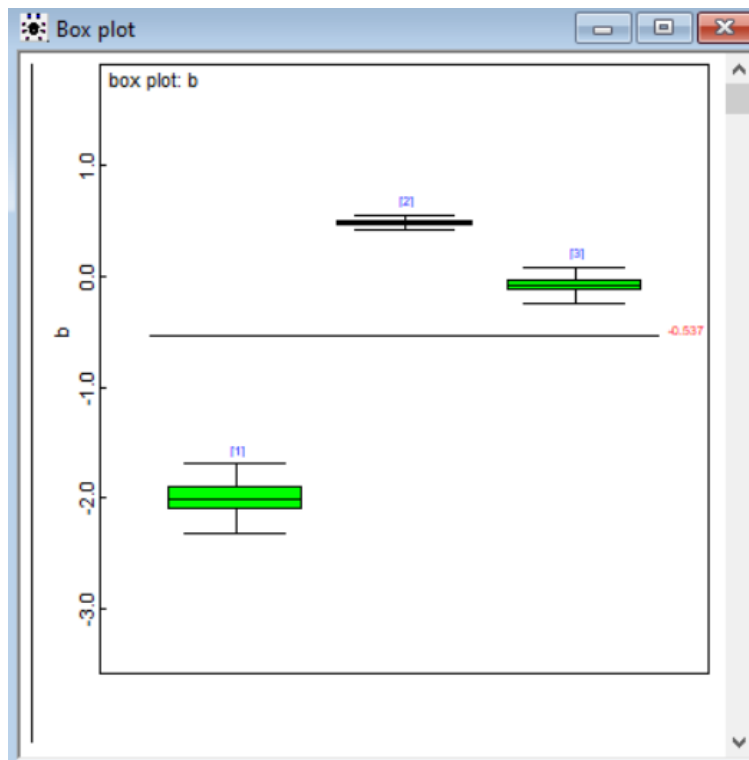


Figure 5: Simulated Results of Model 2

Figure 6: Boxplot of β s in Model 2

3. Model Comparison and Selection:

Model comparison and selection are based on the R^2 , Deviance, DIC, MSE of validating data and Laud-Ibrahim (LI) criteria. The LI is calculated in ProjectLI.odc, see Figure A4, and results are shown in Figure 7. R^2 , Deviance, DIC and MSE are calculated in ProjectMultReg1.odc and ProjectMultReg2.odc. The related results are presented in Figure 3 and 5. All criteria are summarized in Table 1:

Criterion	Model 1	Model 2
R^2	0.8831	0.8811
Deviance	244.1	243.6
DIC	249.2	247.7
MSE of Validating Data	67.53	66.47
LI	54.95	54.55

Table 1: Model Selection Criteria Summary

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
L[1]	54.95	6.312	0.02164	43.92	54.48	68.65	1001	1000000
L[2]	54.55	6.173	0.02177	43.75	54.11	67.96	1001	1000000

Figure 7: LI Criterion Results on Model 1 and 2

R^2 measures how close the training data are to the regression model. Model 1 has 1 more covariate than Model 2, so Model 1 is more complicated and R^2 is higher. As model complexity increases, R^2 increases. However criteria of Deviance, DIC, MSE and LI all show that the Model 2 is better than Model 1, which means that the Model 1 is overfitting.

Conclusion

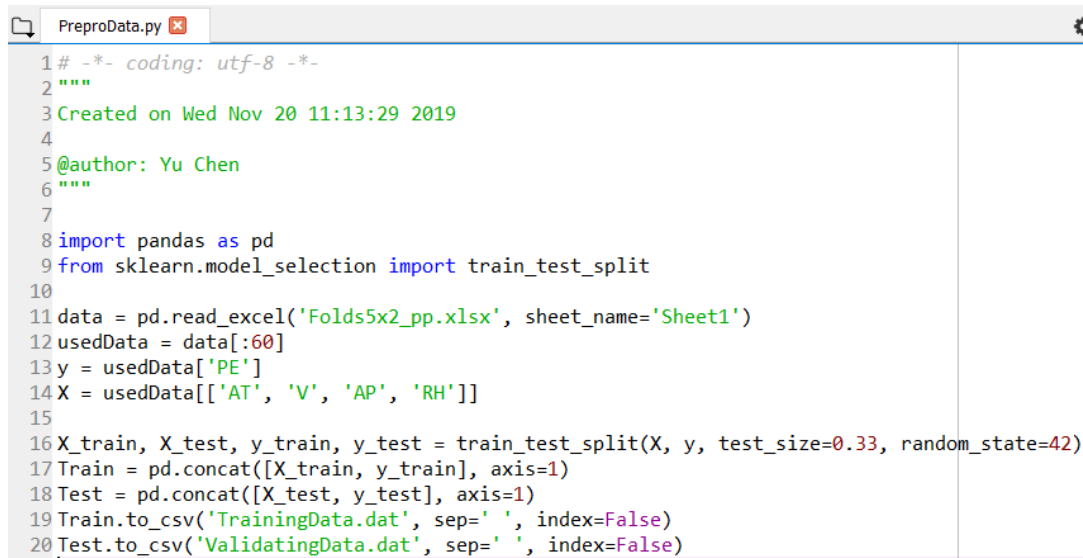
This study presents an alternative solution model for a prediction of the electrical power output of CCPP. The Bayesian multiple regression model is preferred to use for accurate prediction and with less computational time and efforts.

The covariate of Exhaust Vacuum (V) is not significant based on the results of Model 1 and 2. The ambient temperature (AT), atmospheric pressure (AP) and relative humidity (RH) are the most influent covariates in the predictive model using Bayesian multiple regression model.

Reference

1. Pınar Tüfekci, Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods, International Journal of Electrical Power & Energy Systems, Volume 60, September 2014, Pages 126-140, ISSN 0142-0615.
2. Data resource is from UCI machine learning repository:
<http://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant>

Appendix: Code

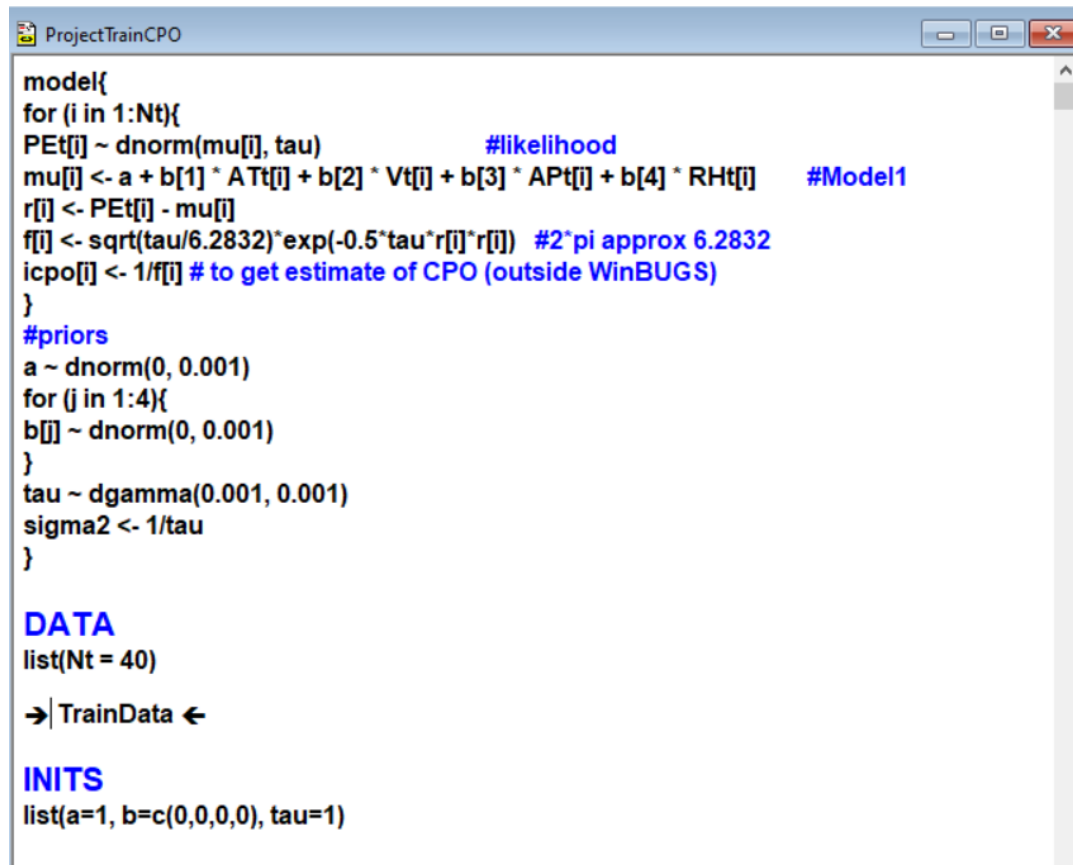


```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Nov 20 11:13:29 2019
4
5 @author: Yu Chen
6 """
7
8 import pandas as pd
9 from sklearn.model_selection import train_test_split
10
11 data = pd.read_excel('Folds5x2_pp.xlsx', sheet_name='Sheet1')
12 usedData = data[:60]
13 y = usedData['PE']
14 X = usedData[['AT', 'V', 'AP', 'RH']]
15
16 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
17 Train = pd.concat([X_train, y_train], axis=1)
18 Test = pd.concat([X_test, y_test], axis=1)
19 Train.to_csv('TrainingData.dat', sep=' ', index=False)
20 Test.to_csv('ValidatingData.dat', sep=' ', index=False)

```

Figure A1: Python Code (ProjectPreprocessData.py) for Data Pre-processing



```

model{
  for (i in 1:Nt){
    PET[i] ~ dnorm(mu[i], tau)          #likelihood
    mu[i] <- a + b[1] * ATt[i] + b[2] * Vt[i] + b[3] * APt[i] + b[4] * RHt[i]  #Model1
    r[i] <- PET[i] - mu[i]
    f[i] <- sqrt(tau/6.2832)*exp(-0.5*tau*r[i]*r[i]) #2*pi approx 6.2832
    icpo[i] <- 1/f[i] # to get estimate of CPO (outside WinBUGS)
  }
  #priors
  a ~ dnorm(0, 0.001)
  for (j in 1:4){
    b[j] ~ dnorm(0, 0.001)
  }
  tau ~ dgamma(0.001, 0.001)
  sigma2 <- 1/tau
}

DATA
list(Nt = 40)
->| TrainData <-

INITS
list(a=1, b=c(0,0,0,0), tau=1)

```

Figure A2: OpenBUGS Code (ProjectTrainCPO.odc) for CPO Analysis on Training Data

The image shows two side-by-side OpenBUGS windows. The left window, titled 'ProjectMultReg1', contains code for a regression model with parameters a , b , and τ . It includes priors, likelihood, and prediction sections. The right window, titled 'ProjectMultReg2', contains similar code but with a different likelihood function and prediction section. Both windows have a 'DATA' section with initial values and an 'INITS' section with starting values.

```

model{
  for (i in 1:Nt){
    PET[i] ~ dnorm(mu[i], tau)
    mu[i] <- a + b[1] * AT[i] + b[2] * VT[i] + b[3] * AP[i] + b[4] * RH[i]
  }
  #priors
  a ~ dnorm(0, 0.001)
  for (j in 1:4){
    b[j] ~ dnorm(0, 0.001)
  }
  tau ~ dgamma(0.001, 0.001)
  #Bayesian R^2
  p <- 5
  nminusp <- Nt - p
  sigma2 <- 1/tau
  sse <- nminusp * sigma2
  for (i in 1:Nt){
    cy[i] <- PET[i] - mean(PET[i])
  }
  sst <- inprod(cy[], cy[])
  BR2 <- 1 - sse/sst
  #prediction
  for (i in 1:Nv){
    newAT[i] <- ATv[i]
    newVT[i] <- VTv[i]
    newAP[i] <- APv[i]
    newRH[i] <- RHv[i]
    ympred[i] <- a + b[1] * newAT[i] + b[2] * newVT[i] + b[3] * newAP[i] + b[4] * newRH[i]
    ypred[i] ~ dnorm(ympred[i], tau)
    diffy[i] <- ypred[i] - PEv[i]
    diffy2[i] <- diffy[i] * diffy[i]
  }
  #MSE
  MSE <- mean(diffy2[])
}

DATA
list(Nt = 38, Nv = 20)
-> CrossValData <-

INITS
list(a=1, b=c(0,0,0,0), tau=1,
ypred=c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0))

```

```

model{
  for (i in 1:Nt){
    PET[i] ~ dnorm(mu[i], tau)
    mu[i] <- a + b[1] * AT[i] + b[2] * AP[i] + b[3] * RH[i]
  }
  #priors
  a ~ dnorm(0, 0.001)
  for (j in 1:3){
    b[j] ~ dnorm(0, 0.001)
  }
  tau ~ dgamma(0.001, 0.001)
  #Bayesian R^2
  p <- 4
  nminusp <- Nt - p
  sigma2 <- 1/tau
  sse <- nminusp * sigma2
  for (i in 1:Nt){
    cy[i] <- PET[i] - mean(PET[i])
  }
  sst <- inprod(cy[], cy[])
  BR2 <- 1 - sse/sst
  #prediction
  for (i in 1:Nv){
    newAT[i] <- ATv[i]
    newVT[i] <- VTv[i]
    newAP[i] <- APv[i]
    newRH[i] <- RHv[i]
    ympred[i] <- a + b[1] * newAT[i] + b[2] * newAP[i] + b[3] * newRH[i]
    ypred[i] ~ dnorm(ympred[i], tau)
    diffy[i] <- ypred[i] - PEv[i]
    diffy2[i] <- diffy[i] * diffy[i]
  }
  #MSE
  MSE <- mean(diffy2[])
}

DATA
list(Nt = 38, Nv = 20)
-> CrossValData <-

INITS
list(a=1, b=c(0,0,0), tau=1,
ypred=c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0))

```

Figure A3: OpenBUGS Code (ProjectMultReg1.odc and ProjectMultReg2.odc) for Regression

The image shows an OpenBUGS window titled 'ProjectLI'. It contains code for a model comparing two models (Model1 and Model2) using the LI criterion. The code includes priors, likelihood, and prediction sections. The 'DATA' section lists the number of observations and the 'INITS' section provides starting values for the parameters.

```

model{
  for (j in 1:Nt){
    mu[1,j] <- a[1] + a[2] * AT[j] + a[3] * VT[j] + a[4] * AP[j] + a[5] * RH[j]
    mu[2,j] <- b[1] + b[2] * AT[j] + b[3] * AP[j] + b[4] * RH[j]
  }
  # compare L from models 1-2
  for (i in 1:2) {
    tau[i] ~ dgamma(0.001, 0.001)
    L[i] <- sqrt(sum(D2[i,j]) + pow(sd(Y.new[i,j]), 2))
  }
  # data sets 1-2 for different models
  for (j in 1:Nt){
    Y[i,j] <- PET[j]
    Y[i,j] ~ dnorm(mu[i,j], tau[i])
    D2[i,j] <- pow(PET[j] - Y.new[i,j], 2)
    Y.new[i,j] ~ dnorm(mu[i,j], tau[i])
  }
  # priors
  for (j in 1:5) {
    a[j] ~ dnorm(0, 0.001)
  }
  for (j in 1:4) {
    b[j] ~ dnorm(0, 0.001)
  }
}

DATA
list(Nt = 38)
-> TrainData <-

INITS
list(a=c(1,0,0,0,0), b=c(1,0,0,0), tau=c(1,1))

For uninitialized variables: GEN INITS

```

Figure A4: OpenBUGS Code (ProjectLI.odc) for LI Criterion