How to predict new compounds

(1) About the models of the prediction system:

The seven models of the multi-layer sweetness prediction system are shown in **Table 1**.

Table 1. Seven models of the multi-layer sweetness prediction system

Category	Natural	Artificial	Carbohydrate	Non-carbohydrate	Nutritive	Non-nutritive	LogSw
Model	MOE2d-XGBoost	MACCS-RF	Atompairs-XGBoost	MOE2d-XGBoost	MOE2d-XGBoost	MOE2d-XGBoost	Atompairs-SVR

(2) Descriptor and fingerprints:

The models were built on KNIME (version 4.1.0). Atompairs (1024 bits), ECFP4 (1024 bits) and MACCS (167 bits) fingerprints were calculated by the RDKit node of KNIME. The MOE2d descriptors were calculated by MOE (version 2018). The details of the selected MOE2d descriptorsrelated to the above seven models are as follows:

- **Natural (105):**

apol, ast_fraglike, ast_fraglike_ext, ast_violation, a_acid, a_aro, a_base, a_donacc, a_ICM, a_nCl, a_nF, a_nN, a_nP, a_nS, balabanJ, BCUT_SLOGP_0, BCUT_SLOGP_2, BCUT_SLOGP_3, BCUT_SMR_0, BCUT_SMR_1, b_1rotN, b_1rotR, b_double, b_max1len, b_triple, chiral, chiral_u, density, diameter, FCharge, GCUT_PEOE_0, GCUT_PEOE_1, GCUT_PEOE_2, GCUT_PEOE_3, GCUT_SLOGP_0, GCUT_SLOGP_1, GCUT_SLOGP_2, GCUT_SMR_0, GCUT_SMR_1, GCUT_SMR_2, h_emd_C, h_logP, h_logS, h_log_dbo, h_log_pbo, h_pavgQ, h_pKa, h_pKb, h_pstates, h_pstrain, Kier3, KierA2, lip_violation, logP(o/w), mutagenic, opr_brigid, opr_leadlike, PEOE_RPC+, PEOE_RPC-, PEOE_VSA+0, PEOE_VSA+1, PEOE_VSA+2, PEOE_VSA+3, PEOE_VSA+5, PEOE_VSA+6, PEOE_VSA-0, PEOE_VSA-1, PEOE_VSA-2, PEOE_VSA-3, PEOE_VSA-4, PEOE_VSA-5, PEOE_VSA_FHYD, PEOE_VSA_FNEG, PEOE_VSA_FPNEG, PEOE_VSA_NEG, PEOE_VSA_PNEG, petitjean, reactive, rsynth, SlogP_VSA1, SlogP_VSA2, SlogP_VSA3, SlogP_VSA4, SlogP_VSA5, SlogP_VSA6, SlogP_VSA7, SlogP_VSA8, SlogP_VSA9, SMR_VSA0, SMR_VSA1, SMR_VSA2, SMR_VSA3, SMR_VSA4, SMR_VSA5, SMR_VSA6, SMR_VSA7, VAdjEq, VAdjMa, VDistEq, vsa_acc, vsa_base, vsa_don, vsa_hyd, vsa_other, weinerPath

- **Non-carbohydrate (124):**

apol, ast_fraglike, ast_fraglike_ext, ast_violation, a_acc, a_acid, a_aro, a_base, a_don, a_ICM, a_nB, a_nBr, a_nCl, a_nF, a_nI, a_nN, a_nP, a_nS, balabanJ, BCUT_SLOGP_0, BCUT_SLOGP_1, BCUT_SLOGP_2, BCUT_SLOGP_3, BCUT_SMR_0, BCUT_SMR_1, BCUT_SMR_2, BCUT_SMR_3, b_1rotN, b_1rotR, b_double, b_max1len, b_triple, chi1_C, chiral, chiral_u, density, diameter, FCharge, GCUT_SLOGP_0, GCUT_SLOGP_1, GCUT_SLOGP_2, GCUT_SLOGP_3, GCUT_SMR_0, GCUT_SMR_1, GCUT_SMR_2, h_ema, h_emd, h_emd_C, h_logD, h_logP, h_logS, h_log_dbo, h_log_pbo, h_pavgQ, h_pKa, h_pKb, h_pstates, h_pstrain, Kier2, Kier3, KierFlex, lip_druglike, lip_violation, logP(o/w), logS, mutagenic, opr_brigid, opr_leadlike, opr_nrot, opr_violation, PEOE_RPC-, PEOE_VSA+0, PEOE_VSA+1, PEOE_VSA+2, PEOE_VSA+3, PEOE_VSA+4, PEOE_VSA+5, PEOE_VSA+6, PEOE_VSA-0, PEOE_VSA-1, PEOE_VSA-2, PEOE_VSA-3, PEOE_VSA-4, PEOE_VSA-5, PEOE_VSA-6, PEOE_VSA_FHYD, PEOE_VSA_FNEG, PEOE_VSA_FPNEG, PEOE_VSA_FPPOS, PEOE_VSA_NEG, PEOE_VSA_PNEG, PEOE_VSA_PPOS, petitjean, reactive, rsynth, SlogP, SlogP_VSA0, SlogP_VSA1, SlogP_VSA2, SlogP_VSA3, SlogP_VSA4, SlogP_VSA5, SlogP_VSA6, SlogP_VSA7, SlogP_VSA8, SlogP_VSA9, SMR_VSA0, SMR_VSA1, SMR_VSA2, SMR_VSA3, SMR_VSA4, SMR_VSA5, SMR_VSA6, SMR_VSA7, TPSA, VAdjEq, VAdjMa, VDistEq, vsa_acc, vsa_base, vsa_don, vsa_hyd, vsa_other, weinerPath

- **Nutritive (102):**

ast_fraglike, ast_fraglike_ext, ast_violation, ast_violation_ext, a_acid, a_aro, a_count, a_don, a_ICM, a_nF, a_nI, a_nN, a_nP, a_nS, balabanJ, BCUT_PEOE_0, BCUT_PEOE_1, BCUT_PEOE_3, BCUT_SLOGP_0, BCUT_SLOGP_2, BCUT_SLOGP_3, BCUT_SMR_2, b_1rotN, b_1rotR, b_double, b_max1len, b_triple, chi0v_C, chi1_C, chiral, chiral_u, density, diameter, GCUT_PEOE_0, GCUT_PEOE_1, GCUT_PEOE_2, GCUT_SLOGP_0, GCUT_SLOGP_1, GCUT_SLOGP_2, GCUT_SMR_0, GCUT_SMR_1, GCUT_SMR_3, h_emd_C, h_logP, h_logS, h_log_dbo, h_log_pbo, h_pavgQ, h_pKa, h_pKb, h_pstates, h_pstrain, Kier3, KierA2, lip_druglike, lip_violation, mutagenic, opr_brigid, opr_leadlike, PEOE_RPC+, PEOE_RPC-, PEOE_VSA+0, PEOE_VSA+1, PEOE_VSA+2, PEOE_VSA+3, PEOE_VSA+5, PEOE_VSA+6, PEOE_VSA-0, PEOE_VSA-1, PEOE_VSA-2, PEOE_VSA-3, PEOE_VSA-4, PEOE_VSA-5, PEOE_VSA_FHYD, PEOE_VSA_FNEG, PEOE_VSA_FPNEG, PEOE_VSA_FPPOS, PEOE_VSA_NEG, PEOE_VSA_PPOS, petitjean, reactive, rsynth, SlogP_VSA1, SlogP_VSA2, SlogP_VSA3, SlogP_VSA4, SlogP_VSA5, SlogP_VSA7, SlogP_VSA8, SlogP_VSA9, SMR_VSA0, SMR_VSA2, SMR_VSA4, SMR_VSA5, SMR_VSA7, TPSA, vsa_acc, vsa_don, vsa_hyd, vsa_other, weinerPath, weinerPol

- **Non-nutritive (122):**

apol, ast_fraglike, ast_fraglike_ext, ast_violation, a_acid, a_aro, a_base, a_don, a_ICM, a_nB, a_nBr, a_nCl, a_nF, a_nI, a_nN, a_nP, a_nS, balabanJ, BCUT_SLOGP_0, BCUT_SLOGP_1, BCUT_SLOGP_2, BCUT_SLOGP_3, BCUT_SMR_0, BCUT_SMR_1, BCUT_SMR_2, BCUT_SMR_3, b_1rotN, b_1rotR, b_double, b_max1len, b_triple, chi1_C, chiral, chiral_u, density, diameter, FCharge, GCUT_SLOGP_0, GCUT_SLOGP_1, GCUT_SLOGP_2, GCUT_SLOGP_3, GCUT_SMR_0, GCUT_SMR_1, GCUT_SMR_2, h_emd, h_emd_C, h_logD, h_logP, h_logS, h_log_dbo, h_log_pbo, h_pavgQ, h_pKa, h_pKb, h_pstates, h_pstrain, Kier2, Kier3, KierFlex, lip_druglike, lip_violation, logP(o/w), logS, mutagenic, opr_brigid, opr_leadlike, opr_nrot, opr_violation, PEOE_PC-, PEOE_RPC-, PEOE_VSA+0, PEOE_VSA+1, PEOE_VSA+2, PEOE_VSA+3, PEOE_VSA+4, PEOE_VSA+5, PEOE_VSA+6, PEOE_VSA-0, PEOE_VSA-1, PEOE_VSA-2, PEOE_VSA-3, PEOE_VSA-4, PEOE_VSA-5, PEOE_VSA-6, PEOE_VSA_FHYD, PEOE_VSA_FNEG, PEOE_VSA_FPNEG, PEOE_VSA_FPPOS, PEOE_VSA_NEG, PEOE_VSA_POL, petitjean, reactive, rsynth, SlogP, SlogP_VSA0, SlogP_VSA1, SlogP_VSA2, SlogP_VSA3, SlogP_VSA4, SlogP_VSA5, SlogP_VSA6, SlogP_VSA7, SlogP_VSA8, SlogP_VSA9, SMR_VSA0, SMR_VSA1, SMR_VSA2, SMR_VSA3, SMR_VSA4, SMR_VSA5, SMR_VSA6, SMR_VSA7, VAdjEq, VAdjMa, VDistEq, vsa_acc, vsa_base, vsa_don, vsa_hyd, vsa_other, vsa_pol, weinerPath

(3) How to form your own prediction pipeline:

The details of constructing your own workflow are shown in **Figure 1** and **Figure2**. Explanation of workflow:

1) The local model file is read by the **Model Reader** node above, while the below reads the model-Normalizer.zip file for normalizer;

2) The node of **File Reader** is used to read the data that needs to be predicted;

3) Convert numbers of label to strings using **Number to String** node is required in

classification models;

4) Select the corresponding prediction node according to the model read by the model reader;

5) Output for the prediction result. In addition, evaluation nodes can be chosen according to your task.

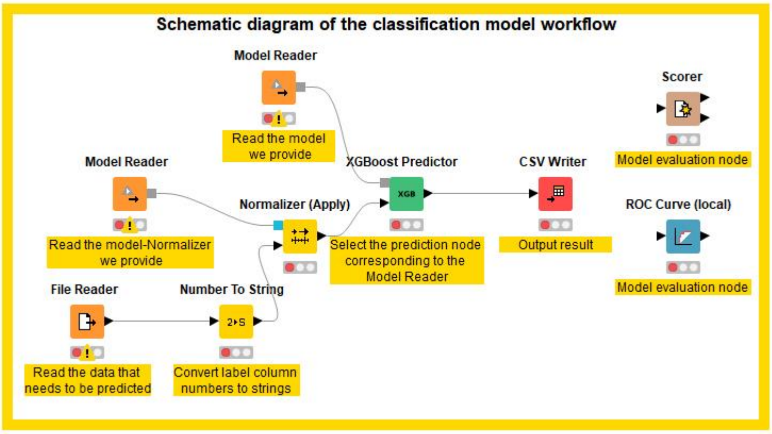

Figure 1. KNIME usage example of classification model.

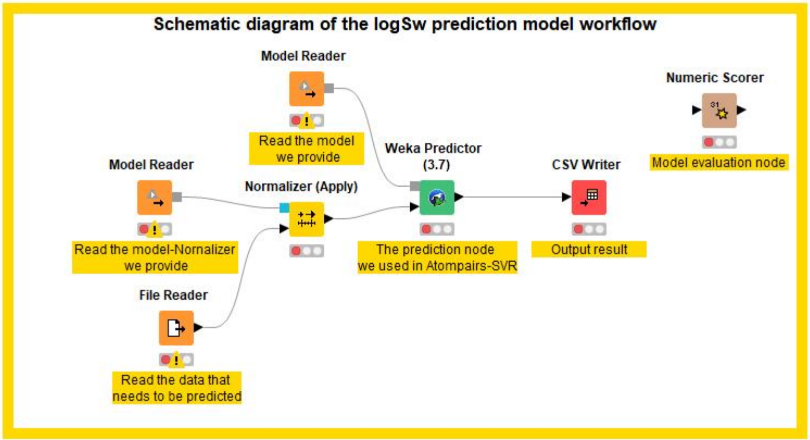

Figure 2. KNIME usage example of logSw prediction model.