DATA11002 Introduction to Machine Learning 2021

Term Project Final Report

Predicting NPF events

Kai Hartzell

December 22, 2021

University of Helsinki

Contents

1. Introduction

In this report, we are studying New Particle Formation (NPF) and predicting NPF events based on training data gathered at the Hyytiälä forestry field station SMEAR II mast on different days in the years 2000-2011 [1, 5].

NPF is an event of particle formation from micro-particles, found in air, to new molecular particles. It is known to affect climate, air quality and thus human and animal health, but the particle formation process is quite complex and not fully known [3, 4].

Therefore, we are using machine learning methods and libraries to help predicting NPF phenomena based on multiple variables found in training data. The interesting question is: When and under which conditions do NPF events occur, and what kind of events could they be? This is also closely related to weather forecasting.

The training data, `npf_train.csv`, contained observed NPF data and exactly 100 feature columns presenting physical characteristics. In fact, there were 50 measured variables in total, and for each of the variables, computed mean and std values. A longer description of the variables can be found at [2].

The measurements have been gathered during various days and times between sunrise and sunset. The test data, `npf_test_hidden.csv`, is similar to the training data, but the classes are unknown and thus need to be predicted.

The class variable `class4` (because of 4 different class categories) could be either `nonevent` or one of the three events: `II`, `Ia` or `Ib`.

The label `nonevent` means that no NPF event occurred, otherwise some of the three events occurred.

Event classes were separated into classes `II` and `I`, the latter into two additional classes `Ia` and `Ib`. Class `II` meant that the confidence level of NPF growth and formation rates was low. Classes belonging to `I` had high confidence level. Class `Ia` meant strong NPF events, while `Ib` included other events [6].

Eventually, I decided to use Gaussian Naive Bayes (NB, or GNB vs. Bernoulli BNB) classifier to predict NPF events based on testing data, `npf_test_hidden.csv`. The "hidden" means that the testing data did not include the class labels.

This project was done using R with R Markdown and Git as version control. The computations as well as tables in this report were performed with R, and the text mostly with R Markdown. The used IDE was RStudio. There exist small displayed chunks of code, but the code is mostly not echoed. The whole project can be found from the Git repo.

In the following sections, I will present the data pre-processing, machine learning methods used, training of the model, discussing and results of the project.

2. Data analysis and pre-processing

The training data, `npf_train.csv`, included 104 columns and 458 observations in total. The test data, `npf_test_hidden.csv`, included 965 unclassified observations. There were columns `id`, `date`, `class4`, `partlybad` and 100 other feature variables measured. The datasets were quite clean and did not need much pre-processing. This shows how the data looked like. This is the first observation.

```
##   id       date   class4 partlybad CO2168.mean CO2168.std CO2336.mean
## 1  1 2000-01-01 nonevent     FALSE     384.462   2.284996    384.1645
##   CO2336.std CO242.mean CO242.std CO2504.mean CO2504.std Glob.mean Glob.std
## 1   2.135062   385.2747  2.211695    383.8851   1.955198  19.24551 11.90955
##   H2O168.mean H2O168.std H2O336.mean H2O336.std H2O42.mean  H2O42.std
## 1    2.278154 0.05150523       2.272 0.05187726   2.316406 0.05165106
##   H2O504.mean H2O504.std H2O672.mean H2O672.std H2O84.mean  H2O84.std NET.mean
## 1    2.262308 0.05589525    2.272769 0.06414115   2.289062 0.05368097 13.96418
##    NET.std NO168.mean  NO168.std NO336.mean  NO336.std NO42.mean   NO42.std
## 1 11.05117 0.07953846 0.07122655 0.08446154 0.06803103  0.054375 0.04730432
##   NO504.mean  NO504.std NO672.mean  NO672.std NO84.mean   NO84.std NOx168.mean
## 1 0.08553846 0.07874069 0.06476923 0.05520155   0.07125 0.06390717    2.142154
##   NOx168.std NOx336.mean NOx336.std NOx42.mean NOx42.std NOx504.mean NOx504.std
## 1  0.7728003    2.136923  0.7141536   2.071875 0.6294565    2.078769  0.6251687
##   NOx672.mean NOx672.std NOx84.mean NOx84.std O3168.mean O3168.std O342.mean
## 1    2.046154  0.5824053   2.085781  0.663547   20.29892  2.258243    19.885
##   O342.std O3504.mean O3504.std O3672.mean O3672.std O384.mean O384.std
## 1 2.231756   20.79462   1.88802   21.31831  1.976654  20.11359 2.273609
##   Pamb0.mean Pamb0.std PAR.mean  PAR.std    PTG.mean     PTG.std RGlob.mean
## 1   999.8595 0.1664669 20.66235 13.77842 0.003560372 0.005574335   7.169164
##   RGlob.std RHIRGA168.mean RHIRGA168.std RHIRGA336.mean RHIRGA336.std
## 1  4.595515       98.33385      1.127238       99.06662      1.324536
##   RHIRGA42.mean RHIRGA42.std RHIRGA504.mean RHIRGA504.std RHIRGA672.mean
## 1      96.47422     1.333359       98.88585      1.375835       101.0309
##   RHIRGA672.std RHIRGA84.mean RHIRGA84.std RPAR.mean RPAR.std SO2168.mean
## 1      1.650608      96.99047     1.117522  8.864799 5.279957    1.070769
##   SO2168.std SWS.mean  SWS.std T168.mean  T168.std  T42.mean   T42.std
## 1  0.1919416 924.8636 12.33769 -12.66211 0.3762745 -12.20161 0.3752439
##   T504.mean  T504.std T672.mean  T672.std  T84.mean   T84.std UV_A.mean
## 1 -12.80858 0.4363618 -13.01647 0.5256979 -12.42297 0.3763239  1.635563
##    UV_A.std  UV_B.mean   UV_B.std   CS.mean       CS.std
## 1 0.8569483 0.02643777 0.01461688 0.0033739 0.0007332531
```

I decided to drop out columns `id`, `date` and `partlybad` from both train and test data, because `id` and `date` did not have any impact on the results, and the value of `partlybad` was always `FALSE`. The `class4` column indicated the observed class, and it was one of these: `II`, `Ia`, `Ib`, `nonevent`.

The binary classification task was to identify nonevent and event classes, ie. `II`, `Ia`, `Ib`. The hidden test data was similar to training data, but did not contain the class values. I replaced the `NA` values with a placeholder `nonevent`. I factored the training data classes as `II`, `Ia`, `Ib`, `nonevent`.

Next, I constructed a data frame and table of the class sd and mean values:

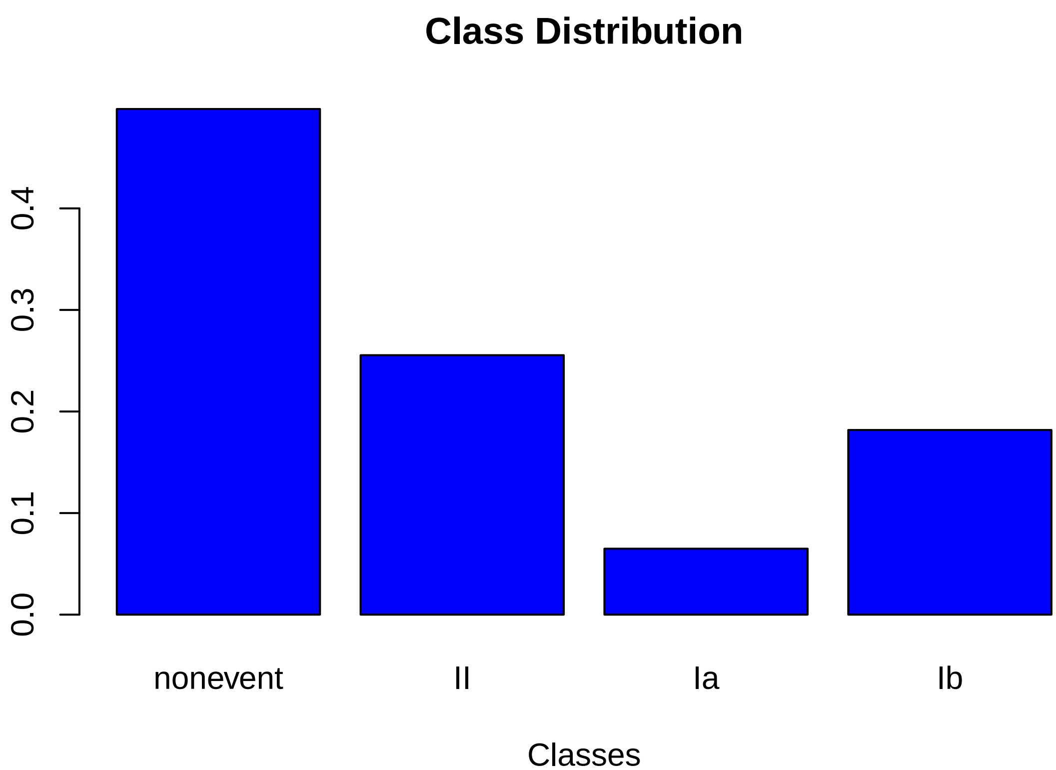
	class_non.mean	class_non.sd	classII.mean	classII.sd	classIa.mean	classIa.sd	classIb.mean	classIb.sd
CO2168.mean	383.3257592	12.0163656	379.7816558	9.6374969	377.3983620	7.9319611	377.5385938	8.5728401
CO2168.std	3.8076389	3.7136048	3.4647404	3.0638298	1.8811666	3.3126275	3.3173097	3.0186016
CO2336.mean	383.3010772	12.0127173	379.8355082	9.6230663	377.4714719	7.8794670	377.6037099	8.5602995
CO2336.std	3.5851381	3.4501180	3.2415884	2.8516257	1.7707188	3.1835412	3.1055142	2.7644834
CO242.mean	384.3454936	11.4029920	380.5672238	9.4285409	378.0545534	7.3849728	378.4111268	8.3496778
CO242.std	4.5978001	4.4815486	4.2714589	3.9259490	2.1576477	3.4830427	4.2127625	4.3486898
CO2504.mean	383.1770183	12.0503130	379.7501748	9.6367019	377.4236330	7.8790896	377.5267348	8.5898615
CO2504.std	3.4192060	3.1583781	2.9971388	2.5841796	1.6631054	3.0425304	2.8382270	2.4519342
Glob.mean	124.1843709	108.2884333	265.4114777	96.6022100	217.4514936	136.5090434	273.8362652	94.1315775
Glob.std	100.3933153	89.5181144	196.3972508	70.2540049	148.3179284	97.8803925	197.6149565	70.4300363
H2O168.mean	8.3952240	4.2909264	6.6839081	3.2319905	4.7256178	3.0865356	6.1144060	2.7852484
H2O168.std	0.5305522	0.4666248	0.6812579	0.4348018	0.3949500	0.3097822	0.6062125	0.3944818
H2O336.mean	8.3234504	4.2414113	6.6003523	3.1875941	4.6750551	3.0275964	6.0381327	2.7308122
H2O336.std	0.5293356	0.4655487	0.6725026	0.4262756	0.3950264	0.3053176	0.6076353	0.3960382
H2O42.mean	8.5241697	4.3895086	6.8414829	3.3113565	4.8100257	3.1670608	6.2500283	2.8805575
H2O42.std	0.5447998	0.4825249	0.7035039	0.4504993	0.4007320	0.3314933	0.6189761	0.3851627
H2O504.mean	8.2824671	4.2084802	6.5496678	3.1622404	4.6486063	2.9945831	5.9924726	2.7011730
H2O504.std	0.5274781	0.4679204	0.6699628	0.4251228	0.3928798	0.3059966	0.6094654	0.3996900
H2O672.mean	8.2507405	4.1830761	6.5114863	3.1356407	4.6267247	2.9575198	5.9550852	2.6803969
H2O672.std	0.5299801	0.4680479	0.6695066	0.4254028	0.3879299	0.3077983	0.6088883	0.4012817
H2O84.mean	8.4704343	4.3504865	6.7679388	3.2783437	4.7685026	3.1332267	6.1851300	2.8420338
H2O84.std	0.5398556	0.4769061	0.6954969	0.4453180	0.3943370	0.3208854	0.6118577	0.3918308
NET.mean	80.5261547	74.3785508	167.8337368	73.0851478	128.5413076	93.6546789	171.6901507	73.6059595
NET.std	87.6830282	78.6126033	174.5705225	62.2764092	133.0591203	82.5527469	172.4932287	64.8169662
NO168.mean	0.0832628	0.1178991	0.0486565	0.0551810	0.0840824	0.1708901	0.0618202	0.0845130
NO168.std	0.0868361	0.0701328	0.0762156	0.0385910	0.0756751	0.0743735	0.1103664	0.1480846
NO336.mean	0.0891754	0.1240921	0.0505879	0.0589467	0.0906982	0.1825832	0.0630157	0.0884501
NO336.std	0.0903346	0.0772049	0.0800398	0.0558533	0.0763280	0.0787067	0.0865659	0.0772434
NO42.mean	0.0708654	0.0952515	0.0372286	0.0413754	0.0679048	0.1414973	0.0495214	0.0668548
NO42.std	0.0985503	0.1232974	0.0735775	0.0415377	0.0749556	0.0684068	0.1128484	0.1676579
NO504.mean	0.0886274	0.1226654	0.0490583	0.0586438	0.0893289	0.1906810	0.0615647	0.0863516
NO504.std	0.0905429	0.0839672	0.0763539	0.0406231	0.0748648	0.0805113	0.0839574	0.0689388
NO672.mean	0.0871272	0.1183312	0.0482930	0.0574997	0.0906560	0.1928020	0.0599156	0.0845935
NO672.std	0.0914093	0.0878955	0.0744356	0.0396351	0.0750297	0.0788534	0.0850076	0.0706539
NO84.mean	0.0696912	0.1038470	0.0393163	0.0464958	0.0749010	0.1535954	0.0525418	0.0760259
NO84.std	0.0797140	0.0603073	0.0729624	0.0419833	0.0751914	0.0719955	0.0944649	0.1160699
NOx168.mean	1.8006764	1.6301601	0.8619501	0.7170419	1.4469960	1.8247297	1.0933323	0.8587382
NOx168.std	0.5197397	0.4535324	0.4625667	0.7003889	0.3706119	0.3411748	0.4621600	0.3910069
NOx336.mean	1.7959607	1.6180526	0.8485011	0.7146270	1.4394845	1.8213682	1.0824423	0.8536414
NOx336.std	0.5499471	0.5687775	0.3930678	0.3140737	0.3676035	0.3367659	0.4216894	0.3590915
NOx42.mean	1.8093522	1.6224146	0.8625380	0.7014339	1.4401144	1.7962648	1.1101887	0.8502164
NOx42.std	0.6384147	0.7791902	0.4397016	0.3326902	0.3895916	0.3643103	0.6164194	0.8641324
NOx504.mean	1.7811794	1.5945210	0.8390183	0.7190962	1.4176531	1.8252514	1.0648497	0.8472283
NOx504.std	0.5897877	0.6577056	0.4328458	0.6036045	0.3526593	0.3312644	0.4132702	0.3432988
NOx672.mean	1.7631446	1.5703224	0.8313492	0.7164193	1.4099343	1.8261845	1.0594163	0.8493411
NOx672.std	0.5487984	0.5379051	0.3815342	0.3356013	0.3555551	0.3316057	0.4136651	0.3522673
NOx84.mean	1.7905142	1.6296561	0.8608148	0.7053100	1.4411304	1.8118328	1.0974465	0.8517585
NOx84.std	0.5127998	0.4434093	0.4490562	0.4547183	0.3744475	0.3504128	0.4846042	0.4074471
O3168.mean	28.9563955	8.7015444	37.1354305	7.9375591	35.5817172	9.5820739	38.1966494	7.6531242
O3168.std	3.5209114	2.3207539	4.0125807	2.2471876	3.1157460	2.1628869	4.0468967	2.3752102
O342.mean	27.7531718	8.5799124	35.9105184	8.1806544	34.7646199	9.6125137	37.1545134	8.0531688

	class_non.mean	class_non.sd	classII.mean	classII.sd	classIa.mean	classIa.sd	classIb.mean	classIb.sd
O342.std	3.9320093	2.5715429	4.5629995	2.5121122	3.4300530	2.2983362	4.5916329	2.6110586
O3504.mean	29.9223688	8.7826482	38.1124111	7.6215776	36.1765368	9.4885172	38.9206969	7.4458559
O3504.std	3.3379269	2.1808745	3.6198586	2.0168362	2.9581700	2.1895254	3.6856651	2.1572885
O3672.mean	30.2570159	8.8153168	38.4638542	7.5423407	36.4179137	9.4671344	39.1708386	7.4083422
O3672.std	3.2974286	2.1320519	3.4793625	1.8999729	2.8855798	2.2111996	3.5270694	2.0859228
O384.mean	28.2950117	8.6449395	36.4847757	8.0918533	35.2079586	9.5844538	37.6702692	7.8834566
O384.std	3.6818176	2.4325495	4.2321932	2.3653424	3.1852934	2.1710969	4.2537071	2.4787015
Pamb0.mean	989.5739182	10.4595733	992.4344221	7.5447690	993.0831075	12.7955733	993.8398014	9.2653530
Pamb0.std	0.9036629	0.7414164	1.1832659	0.8141691	1.0277771	0.6889135	1.1659888	0.8041929
PAR.mean	252.1034293	219.8362910	521.8252647	189.1382262	415.7491902	262.7742736	533.1584717	184.6378523
PAR.std	201.6516309	180.6231724	387.9777895	139.0148940	286.6517870	191.6229248	386.9933144	139.2281505
PTG.mean	0.0013185	0.0072903	-0.0005988	0.0047814	-0.0007118	0.0053309	-0.0016611	0.0031918
PTG.std	0.0067485	0.0064188	0.0120327	0.0058381	0.0102281	0.0069848	0.0115483	0.0054228
RGlob.mean	18.2423426	14.4447614	36.2972617	12.9405799	32.9937881	16.9661898	38.3190568	11.9587230
RGlob.std	13.7137875	10.0052002	23.9939113	6.5384811	20.5618458	9.8098910	24.4001760	6.4951044
RHIRGA168.mean	79.9616469	15.9356044	57.2879848	15.6409709	60.9485167	19.8261856	55.8371629	13.4217013
RHIRGA168.std	5.5691925	4.7117565	11.6976471	4.5173211	9.6863927	6.1905295	12.1942132	4.0480959
RHIRGA336.mean	80.4897203	16.2525874	57.4444120	15.8227254	61.3027904	20.0627248	56.0733062	13.6805601
RHIRGA336.std	5.5792926	4.6981200	11.4247132	4.4972056	9.5752945	6.1246428	11.9648446	4.0725800
RHIRGA42.mean	80.3492081	15.1864570	58.4772003	15.3820059	61.1773670	18.9236909	56.8580399	13.2335082
RHIRGA42.std	5.7424427	5.0783234	12.4831364	4.7587657	9.9832382	6.4979431	13.0917220	4.2715860
RHIRGA504.mean	80.5896461	16.3846137	57.3573156	15.8184975	61.3350061	20.1122624	56.1103848	13.8601198
RHIRGA504.std	5.5425971	4.6848747	11.1521853	4.4914608	9.2819531	6.0207258	11.5776651	4.1202661
RHIRGA672.mean	81.4285506	16.7576380	57.8459091	16.1270668	62.0233782	20.3404874	56.6574265	14.2507381
RHIRGA672.std	5.5541586	4.6333869	10.9344953	4.5554081	9.1514718	5.9629777	11.2599693	4.2834540
RHIRGA84.mean	80.0840781	15.5444730	57.6708747	15.5608801	61.0015883	19.5211092	56.0668416	13.3454606
RHIRGA84.std	5.7412174	4.9777942	12.1984752	4.6677814	9.9443157	6.3240611	12.7443155	4.1544103
RPAR.mean	14.1556559	12.5023017	22.5765995	13.2180729	23.3232753	11.0389199	24.4094019	9.8784467
RPAR.std	10.5616485	8.7993333	15.8318446	7.3983849	15.2332906	6.8970832	16.9022109	6.0921322
SO2168.mean	0.2969173	0.4872757	0.1926844	0.1883541	0.1695263	0.1507744	0.2362119	0.3058720
SO2168.std	0.1529806	0.1386084	0.1577387	0.1276472	0.1289616	0.0805043	0.1684688	0.1246219
SWS.mean	901.2928793	39.4040349	915.2222334	18.7440739	923.1286425	9.5964837	919.5428844	13.0544876
SWS.std	28.9825589	43.5319573	16.4777475	35.0055353	5.8773344	17.8563535	12.6442312	27.7441377
T168.mean	6.0779466	10.9514407	8.5962921	8.1833693	2.8687104	8.3095384	7.6928053	8.0038911
T168.std	1.3599286	0.9772331	2.3834288	0.8861056	2.0180805	1.1075181	2.4789441	0.9679934
T42.mean	6.1653770	10.9224699	8.6499311	8.2085590	2.9865494	8.2421630	7.7541718	8.0064785
T42.std	1.4646568	1.1112941	2.6472802	1.0024390	2.1859501	1.2825396	2.7393903	1.0755007
T504.mean	5.8160029	10.8954991	8.2703335	8.2021538	2.5531536	8.2892509	7.3368535	8.0393041
T504.std	1.2650820	0.9053966	2.1760562	0.8340721	1.8472449	1.0164464	2.2836336	0.9179909
T672.mean	5.6329681	10.8540661	8.0609689	8.1864121	2.3391703	8.2753919	7.1069544	8.0400739
T672.std	1.2171312	0.8754616	2.0874301	0.8033751	1.7835714	0.9649712	2.1963858	0.8942363
T84.mean	6.1500669	10.9556256	8.6976018	8.2085735	2.9538699	8.2793216	7.8076448	8.0135334
T84.std	1.4406672	1.0637270	2.5498165	0.9428052	2.1481955	1.1791491	2.6451795	1.0221479
UV_A.mean	7.6842886	6.1089685	14.5738046	5.0739380	11.3026124	6.9411607	14.7907621	5.0123836
UV_A.std	5.6597847	4.8468768	10.3911167	3.9771654	7.5590838	5.1088458	10.3899427	3.9910193
UV_B.mean	0.3258228	0.3044106	0.6028462	0.2732618	0.4214711	0.2929315	0.5940769	0.2781953
UV_B.std	0.2887586	0.2824756	0.5214276	0.2522659	0.3465650	0.2574466	0.5066760	0.2556018
CS.mean	0.0037101	0.0026296	0.0024940	0.0015237	0.0017976	0.0013724	0.0024524	0.0016285
CS.std	0.0006865	0.0005781	0.0006405	0.0006305	0.0004566	0.0004533	0.0006782	0.0004962

We notice that there are some undefined measurements, but they do not have a large impact. Laplace smoothing of 1 was used for the data. The estimated class probabilities for training data:

```
nb_class
```

```
##
##   nonevent         II         Ia         Ib
## 0.49783550 0.25541126 0.06493506 0.18181818
```



The class distibution plotted.

3. Machine learning methods and steps

I applied Gaussian NB classifier [7] to compute the class probabilities for all rows of testing data. The variables were considered as conditionally independent (because of that is the pre-assumption), even though some of them might have a relationship, smaller or larger correlation.

The variables were studied and only a subset of the variables were used. There were some challenges choosing the variables. The classifier predicted the probabilities of each class for each row. The row was identified as `nonevent`, if the probability was higher than the `nb_class` probability for that class.

The formula of NB Gaussian density was

$$\frac{e^{(-(x-\mu)^2/(2*\sigma^2))}}{\sqrt{2*\pi*\sigma^2}} \tag{1}$$

I used some small coefficient adjustments and modifications for the multiclass classification problem. The multiclass problem was, of course, more challenging than the binary classification problem because of more events to be predicted.

The predicted probabilities were compared step by step, with different coefficients. The target was to produce reasonable prediction probabilities for the classes.

After having identified the class as nonevent or event, the event class had to be predicted, if it was not nonevent. I compared the probabilities of different predicted events with NB classifier and chose suitable coefficients for predicting the event classes. I guessed that the accuracy of the binary classification could be 0.73.

Regarding the different classification sub-tasks, the main target was to build a reasonably performing binary classifier. Building a more accurate event class classifier was not as important, although the higher accuracy, the better.

The first 15 estimated classes and probabilities were:

```r
head(df, 15)
```

```
##
## 1      0.73
## 2    class4                   p
## 3        Ia    0.727504450479219
## 4  nonevent    0.080463843559317
## 5        Ib    0.990897111743513
## 6        II    0.991238939899939
## 7  nonevent    0.120253990842419
## 8        II    0.978951849227886
## 9  nonevent 0.000677548460571997
## 10 nonevent  0.00245307701477748
## 11       II    0.926588565138545
## 12       II    0.539002979107427
## 13 nonevent  0.00085019262958641
## 14 nonevent    0.497539389391826
## 15       Ia    0.790420964630693
```

The first row contains the guessed accuracy, and the second row labels for the classes and probabilities: `class4` and `p`.

After that, each row contains the predicted class and prediction probability for the class being an event class, for each data row in the test data. So if the probability `p` is 0.3, the probability of a `non-event` would be 0.7.

This whole data frame was exported as a csv file `answers.csv`. That file contains all the predicted results.

4. Summary and discussing

The predicted class distributions for testing data for classes `nonevent`, `II`, `Ia`, `Ib`:

```
## [1] 0.4611399
```

```
## [1] 0.2683938
```

```
## [1] 0.09948187
```

```
## [1] 0.1709845
```

These probabilities were quite close to the probabilities in training data.

It turned out that my binary accuracy (predicting event vs. nonevent) was about 0.795, actually higher than the guessed 0.73. The margin of error was quite small: about 0.065. In addition, the multiaccuracy (predicting the correct class) was around 0.585, meaning that the model could have performed better, but at least it predicted almost 3 out of 5 classes correctly. As expected, it was more difficult to predict event classes than to predict binary classes (i.e. event vs. non-event).

The perplexity turned out to be 1.65, meaning that it was placed between 1 and 2, the perplexities of "perfect" and dummy random classifiers. The perfect classifier would predict the class always correct, and dummy classifier would assign probability 0.5 to both binary classes.

Regarding the methods, I considered using also cross-validation, Random Forest, logistic regression, kNN and SVM. I ended up in using NB, because of the pros of it. They are that the model is highly scalable and simple generative classifier, it can usually be trained efficiently in supervised learning, and it often requires only a small number of training data. In addition, it is not much affected by random noise and rarely leads to overfitting.

The downsides of using NB in this project were that all the variables were not independent and normally distributed. Those are the assumptions, but by leaving the correlated and not normally distributed variables out, the assumption is still holding and the classifier performed reasonably. The model can still be biased in some situations.

One of the features of NB is that it can be making strong assumptions based on the data, because it is a "naive" model. However, that can also be advantageous, but not every time. Moreover, the model requires information about the distributions and the probabilities, which need to be estimated.

As a hindsight, different subset of the feature variables could have been used. One option would have been to use e.g. only mean values instead of std values, or other variables based on the correlations.

After making some tweaks, I got the NB classifier to predict reasonable results, but there were initially some problems with the class distributions. I learned a lot about the effectiviness and usability of the NB classifier. Hopefully this research is helpful in some way and makes opportunities for future work.

5. Self-grading

"At the end of the course, you will be asked to give your project deliverables (final report, presentation, and challenge submission) an integer grade on a scale from 0 (fail) to 5 (excellent)".

I am giving these grades to my deliverables:

- Challenge submission: 3. This looked as it should, and was returned in time. The accuracy could have been a bit better, but there are no large problems with it. This showed understanding of the topic and was suitable for the problem. The challenge was based on the model.
- Presentation: 0. Unfortunately, I didn't have time with this.
- Final report: 3. The level of final report was about the same as in challenge submission. Nothing relevant was missing, and this showed some deeper understanding of the topic as well as critical analyzing. The readability should be ok. However, the report could have been a bit more comprehensive, and more machine learning methods could have been used. All in all, the topics and research questions were answered in sufficient manner.

The average grade of these deliverables is 2, so I will give myself a grade 2 of this project in total. The minimal requirements are satisfied; the presentation was not defined as a minimal requirement anywhere. But unfortunately, that drops the grade with a number. The work mostly follows the instructions given.

References

[1] https://www2.helsinki.fi/en/research-stations/hyytiala

[2] https://wiki.helsinki.fi/pages/viewpage.action?pageId=243959901

[3] https://iopscience.iop.org/article/10.1088/1748-9326/aadf3c

[4] https://www.who.int/health-topics/air-pollution

[5] http://www.borenv.net/BER/archive/pdfs/ber10/ber10-323.pdf

[6] https://acp.copernicus.org/articles/18/9597/2018/

[7] https://iq.opengenus.org/gaussian-naive-bayes/

Git repo

https://github.com/hartzka/iml21