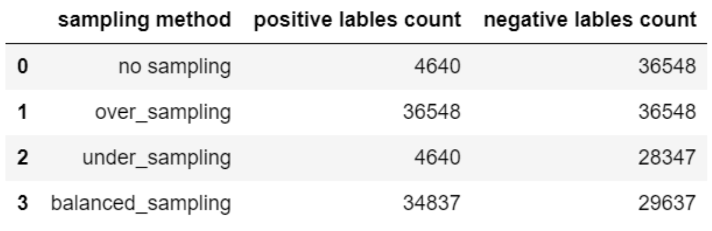
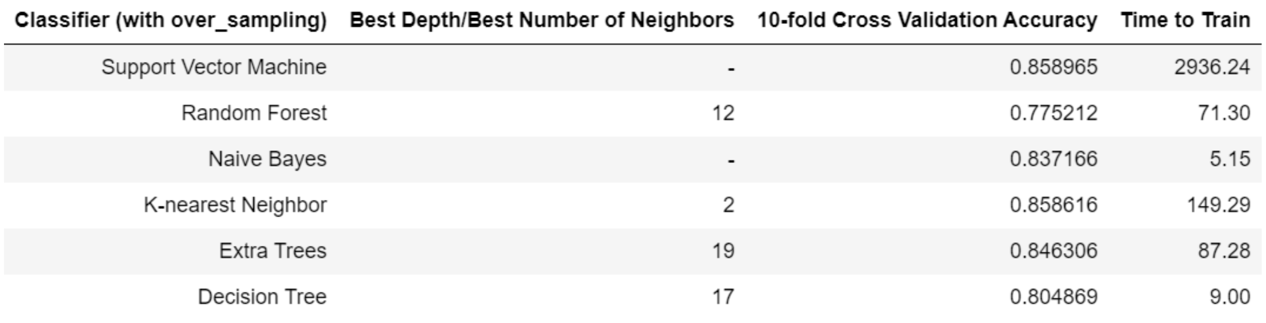
Part1

Three rebalancing methods (over_sampling, under_sampling, balanced_sampling) were used to decrease the class imbalance problem. Based on the count of positive and negative samples after each rebalancing method, banalced_sampling seems better and mode reasonable. The results of classification will also confirm that balanced_sampling is the best for our task:

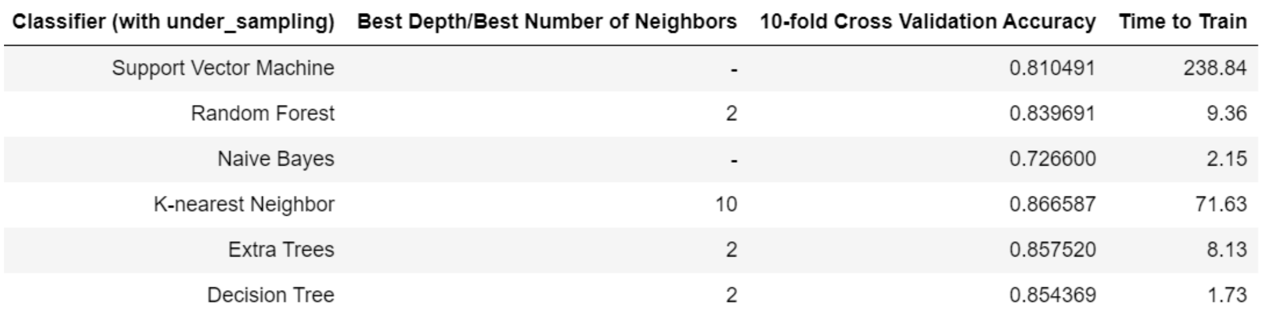


	sampling method	positive lables count	negative lables count
0	no sampling	4640	36548
1	over_sampling	36548	36548
2	under_sampling	4640	28347
3	balanced_sampling	34837	29637

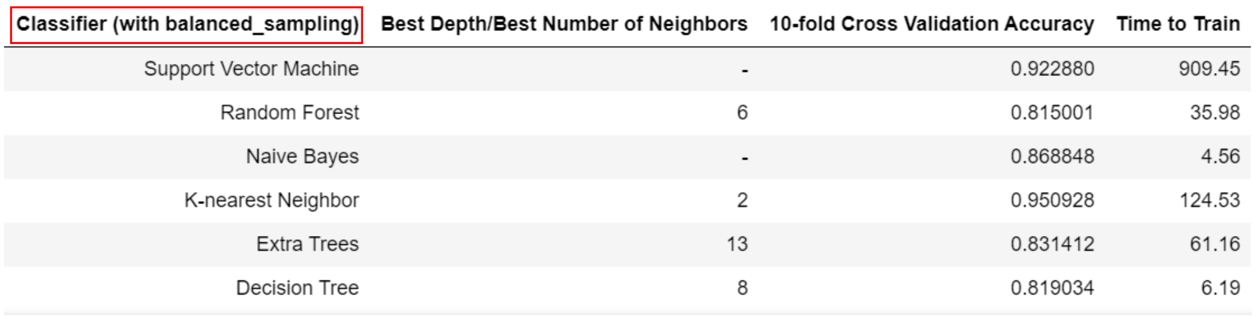
Then, 6 classifiers were built based on the results of each rebalancing method. Models were built by trying different hyperparameters and the best ones were selected (e.g. the best depth for decision tree, and the best number of neighbors for KNN). Below are the results:



Classifier (with over_sampling)	Best Depth/Best Number of Neighbors	10-fold Cross Validation Accuracy	Time to Train
Support Vector Machine	-	0.858965	2936.24
Random Forest	12	0.775212	71.30
Naive Bayes	-	0.837166	5.15
K-nearest Neighbor	2	0.858616	149.29
Extra Trees	19	0.846306	87.28
Decision Tree	17	0.804869	9.00



Classifier (with under_sampling)	Best Depth/Best Number of Neighbors	10-fold Cross Validation Accuracy	Time to Train
Support Vector Machine	-	0.810491	238.84
Random Forest	2	0.839691	9.36
Naive Bayes	-	0.726600	2.15
K-nearest Neighbor	10	0.866587	71.63
Extra Trees	2	0.857520	8.13
Decision Tree	2	0.854369	1.73



Classifier (with balanced_sampling)	Best Depth/Best Number of Neighbors	10-fold Cross Validation Accuracy	Time to Train
Support Vector Machine	-	0.922880	909.45
Random Forest	6	0.815001	35.98
Naive Bayes	-	0.868848	4.56
K-nearest Neighbor	2	0.950928	124.53
Extra Trees	13	0.831412	61.16
Decision Tree	8	0.819034	6.19

Based on above cross validation accuracy results, it seems that balanced_sampling has the best results. Especially, SVM (92.2%) and KNN (95%) have very good results with balanced_sampling method compared to the other rebalancing methods. Naïve-Bayes model also performs better with balanced_sampling. However, tree_based models (DT, RF, Extra Trees) perform better with under_sampling method. **Overall, balanced_sampling is chosen as the best for the rest of the assignment.**

Also, considering the results of balanced_sampling, KNN and SVM are selected as the best models having the best cross validation accuracy. Although their execution time is higher than the other models, but their accuracy is far better than the other models and their execution time is also still reasonable. So, SVM and KNN are considered as the best two models.

Afterwards, the accuracy of each classifiers against each folds was calculated and the paired t_test was done to find if the difference between classifiers are due to chance or there is really a significant difference between them. The final results of t-test show that only KNN-NB, KNN-RF, and KNN-Extrees have significant difference with one another. For the rest of the classifiers, the null hypothesis is accepted, and we can say they have performed equally.

<u>Note</u>: Confidence level of 95% (alpha=0.05) was considered for t-test. All the tables and results are visible in the code.

Fold	DT-NB	DT-KNN	DT-SVM	DT-RF	DT-ExTree	NB-KNN	NB-SVM	NB-RF	NB-Extree	KNN-SVM	KNN-RF	KNN-Extree	SVM-RF	SVM-Extree	RF-Extree
1	-0.152916	-0.217122	-0.218362	-0.018921	-0.196185	-0.064206	-0.065447	0.133995	-0.043269	-0.001241	0.198201	0.020937	0.199442	0.022177	-0.177264
2	-0.081731	-0.172301	-0.182847	-0.049007	0.202233	-0.090571	-0.101117	0.032723	0.283964	-0.010546	0.123294	0.374535	0.133840	0.385081	0.251241
3	0.130893	-0.021867	-0.031483	0.003722	-0.029777	-0.152761	-0.162376	-0.127171	-0.160670	-0.009615	0.025589	-0.007909	0.035205	0.001706	-0.033499
4	-0.383995	-0.393145	-0.583127	0.051954	0.013027	-0.009150	-0.199132	0.435949	0.397022	-0.189981	0.445099	0.406172	0.635081	0.596154	-0.038927
5	-0.136653	-0.447805	-0.347293	0.025748	-0.129362	-0.311152	-0.210641	0.162401	0.007290	0.100512	0.473554	0.318443	0.373042	0.217931	-0.155111
6	0.069645	-0.017062	0.088568	-0.004653	0.024663	-0.086707	0.018924	-0.074298	-0.044982	0.105631	0.012409	0.041725	-0.093222	-0.063906	0.029316
7	0.013650	-0.016287	0.059718	0.005739	-0.002792	-0.029936	0.046068	-0.007911	-0.016442	0.076004	0.022026	0.013495	-0.053979	-0.062510	-0.008531
8	0.015821	-0.008531	0.061579	0.010082	0.001861	-0.024352	0.045758	-0.005739	-0.013960	0.070110	0.018613	0.010392	-0.051497	-0.059718	-0.008221
9	0.012564	-0.012719	0.056615	0.006980	-0.005274	-0.025283	0.044051	-0.005584	-0.017838	0.069335	0.019699	0.007445	-0.049635	-0.061889	-0.012254
10	0.014580	-0.012099	0.058167	0.008686	-0.002172	-0.026679	0.043586	-0.005894	-0.016752	0.070265	0.020785	0.009927	-0.049480	-0.060338	-0.010858
*avgerage	-0.049814	-0.131894	-0.103847	0.004033	-0.012378	-0.082080	-0.054032	0.053847	0.037436	0.028047	0.135927	0.119516	0.107880	0.091469	-0.016411
*standard deviation	0.132859	0.153557	0.203761	0.023843	0.093565	0.082874	0.097289	0.144068	0.152278	0.079846	0.163638	0.155652	0.214095	0.210419	0.104367
*p-value	0.419704	0.054750	0.248358	0.438969	0.902856	$\boxed{0.030758}$	0.205544	0.174422	0.318508	0.194955	$\boxed{0.013866}$	$\boxed{0.019903}$	0.088366	0.127749	0.845297

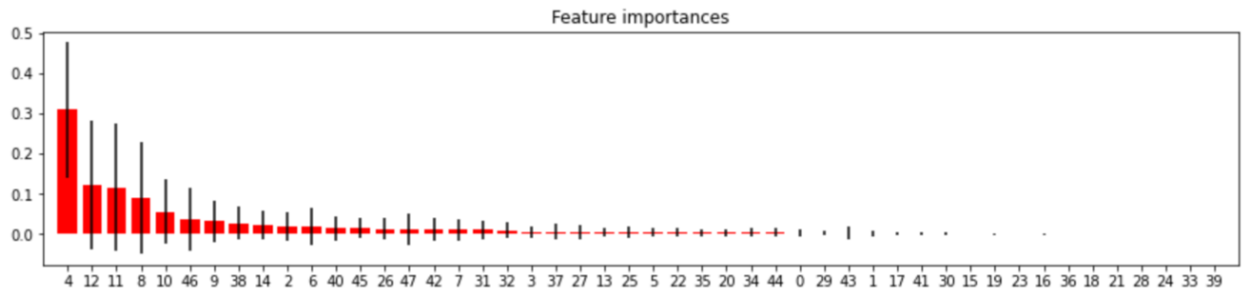
Next, I used two feature selection strategies, i.e. **filter method** and **wrapper method** to see how they affect the models.

For filter method, I used univariate feature selection strategies, that removes all but the k highest ranked features. I tried this method with different k. Chi2 was considered as scoring method. Below are the results on SVM and KNN models. For KNN, 30 selected features have resulted in the best accuracy although for 20 features also the accuracy is very close to the maximum. Also, the training time decrease as the number of features decrease. However, for KNN, the best result is obtained when using all the features.

number os selected features (filter method)	10-fold Cross Validation Accuracy on SVM model	Time to Train
30	0.948102	310.45
20	0.941650	280.14
40	0.940128	307.77
all(48)	0.922880	909.45
10	0.677047	788.55

number os selected features (filter method)	10-fold Cross Validation Accuracy on KNN model	Time to Train
all(48)	0.950928	124.53
40	0.920834	662.50
30	0.914165	529.74
20	0.902503	224.61
10	0.628841	154.38

For wrapper method, I used a random forest model that rank the features by building a RF model. Below results show the importance of all features based on their index in dataset.



Below are the feature names of the top ten high ranked features.

```
duration
nr.employed
euribor3m
emp.var.rate
cons.conf.idx
poutcome_nonexistent
cons.price.idx
default_unknown
job_blue-collar
month
```

I chose 30 top ranked features for the classification and below are the results. It shows that both SVM and KNN have better performed when considering all the features. However, the training time has decreased with feature selection. So, for SVM for example, since the accuracy of the model with feature selection is very close to all features, we might choose 30 features for classification as it results in a faster training.

	number os selected features (wrapper method)	10-fold Cross Validation Accuracy on SVM model	Time to Train
1	all(48)	0.92288	909.45
0	30	0.91877	746.96

	number os selected features (wrapper method)	10-fold Cross Validation Accuracy on KNN model	Time to Train
1	all(48)	0.950928	124.53
0	30	0.927896	88.23

Part2

4 datasets were used as asked on the question. As preprocessing step, all null values of datasets were replaced. The null values of categorical features were replaced with “unknown” and the null values of numerical features replaced with mean value.

Then, 3 algorithms, i.e. KNN, SVM, RF were used to build the classifiers on 4 datasets. Again, different hyperparameters were tried and the best ones selected. All models with different hyperparameters are illustrated in notebook code. The results show that KNN has the best results on bank marketing dataset compared to SVM and RF, while SVM has the best results on labor relation and iris datasets. RF has the best results on congress voting dataset.

Dataset	KNN	SVM	RF
Labor Relations	0.925000	0.97500	0.950000
Congress Voting	0.933245	0.95851	0.960782
Iris	0.940000	0.96000	0.946667
Portuguese Bank Marketing	0.950928	0.92288	0.815001

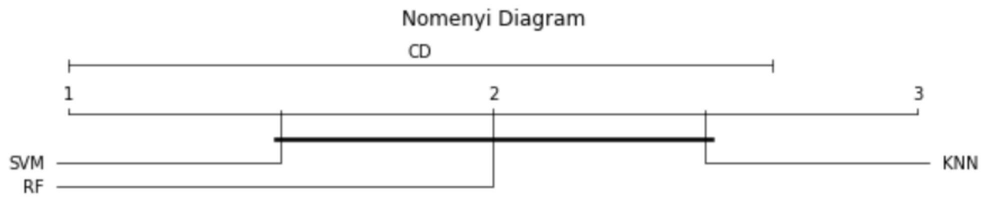
In order to compare the classifiers to find if they really perform differently or their difference is due to chance , considering that we have 4 different datasets, Friedman test was done. Below are the ranks of classifiers against each dataset (1 is the highest rank and 3 is the lowest rank).

Dataset	KNN Rank	SVM Rank	RF Rank
Labor Relations	3.0	1.0	2.0
Congress Voting	3.0	2.0	1.0
Iris	3.0	1.0	2.0
Portuguese Bank Marketing	1.0	2.0	3.0
Avg Rank	2.5	1.5	2.0

Based on the final results of Friedman test, the null hypothesis is accepted, and we can say all classifiers perform equally.

Note: Confidence level of 95% (alpha=0.05) was considered for Friedman test. All the tables and results are visible in the code.

Below Nemenyi diagram also confirm that there is no significant difference between any pair of classifiers as the difference between their average rank is less than the critical value illustrated at the top of the diagram.



The details of all parts, figures and tables are available at notebook python code.