Atlantbh, Internship, Data Science project
Mehmed Kadrić

PREPROCESSING

Firstly, it's important to analyze given dataset. I've found missing values in *Age* column and one missing value in *Pclass* column. I didn't find any outliers except one in *Pclass* column. It's possible to reduce a number of features. Instead of *Name* and *Sex* column it's possible to put *Title* column. Instead of numbers in *Age* column, it's possible (acording to expert knowledge) to form new cathegorical variable with five states: child, teen, adult, mature and old.

CLASSIFICATION

Response variables in machine learning models can be either *quantitative* or *qualitative*. Qualitative variables are often called *cathegorical* variables. For example, eye color is qualitative, taking on values blue, brown, or green. Approaches for predicting qualitative (cathegorical) responses are known as classification process.
I've found several classification algorithms: Logistic Regression, Naive Bayes classifier, Support Vector Machines, K-nearest neighbor, Decision Trees, Neural Networks, etc.
Considering the given dataset (*Titanic*) I realized that my dataset is realitvely small as well as the number of features. Futhermore, it's a binary classification problem where probability of passenger status (survived or not) should be given. Therefore I can make a decision which algorithm to use acording to these facts and properties of earlier enumerated classification algorithms.

Logistic regression

Pros:	**Cons:**
Probability as an output	Doesn't perform well when feature space is too large
Efficient implementations across tools	Doesn't handle large number of categorical features/variables well
Wide spread industry comfort	

Naive Bayes classifier

Pros:	**Cons:**
Computationally fast	Relies on independence assumption and will perform badly if this assumption is not met
Simple to implement	
Works well with high dimensions	

Support Vector Machines

Pros:	**Cons:**
Can handle large feature space	Not very efficient with large number of observations
Can handle non-linear feature interactions	It can be tricky to find appropriate kernel sometimes
Do not rely on entire data	

K-nearest neighbor

Pros:	Cons:
Simple to implement	**Sensitive to class-outliers**
Naturally handle multi-class cases	Computationally expensive
Almost no asssumptionts about the data	In case of many dimensions, the effectiveness of KNN is reduced
Ability to learn complex non-linear decision boundaries	

Decision Trees

Pros:	Cons:
Data classification without much calculations	**High classification error rate while training set is small**
Dealing with noisy or incomplete data	Non-robust – a small change in the data can cause a large change in the final estimated tree
Easy to explain	

Neural Networks

Pros:	Cons:
Good behaviour with nonlinear data with large number of inputs	**A lot of data is needed**
Once trained, the predictions are pretty fast	

I've decided to implement three methods: Linear Regression, Naive Bayes and SVM. The reasons why another three methods aren't implemented are bold and shown in the tables above (*Cons* column).

Comparison

A way to measure quality of a model is sklearn method called *classification_report*. Classification reports for each implemented algorithm are given below.

Table 1 – Logistic Regression report

	Precision	Recall	F1-score	Support
0	0.75	0.87	0.80	255
1	0.64	0.45	0.53	136
Avg/total	0.71	0.72	0.71	391

Accuracy: 72.12%.

Table 2 – Support Vector Machine report

	Precision	Recall	F1-score	Support
0	0.84	0.85	0.84	265
1	0.67	0.65	0.66	126
Avg/total	0.78	0.78	0.78	391

Accuracy: 78.26%.

Table 3 – Naive Bayes report

	Precision	Recall	F1-score	Support
0	0.78	0.82	0.80	264
1	0.58	0.51	0.54	127
Avg/total	0.71	0.72	0.71	391

Accuracy: 71.87%.

Interpretation of results (Logistic Regression is used as an example):
Accuracy: 72.12% of the model's predictions are correct.
Precision: Precision is measure of how precise the model's predictions are. When the model predicts a passenger survived, that person actually did survive **71%**.
Recall: Also called *Sensitivity*. If there a passenger that survived is present in the best set, the model is able to identify (recall) it **72%**.
F1 Score: F1 Score is the best of both worlds as it is weighted average of precision and recall. An F1 Score of ***71%*** means that 71% of time: when the model predicts someone survived you can be confident that person actually did survive and it is not a false alarm, when there is an actual survivor in the dataset, the model is able to detect it in classification problems where there are more than two labels that apply, accuracy is less intuitive and the F1 Score is better measure of a model's performance.
In considered algorithms, Precision gaves us the information of all passengers that we predicted as *survived*, what fraction actually survived. Recall gives us the information of all passengers that actually survived, what fraction did we correctly detect as survived.

PARAMETER TUNING WITH GridSearch

This idea of creating a 'grid' of parameters and just trying out all the possible combinations is called a Gridsearch, this method is common enough that Scikit-learn has this functionality built in with GridSearchCV (CV stands for cross-validation).
The results after parameter tuning are shown in tables below.

Table 4 – Logistic Regression with tuned parameters: C=1.1, penalty: 'l1'

	Precision	Recall	F1-score	Support
0	0.73	0.88	0.80	249
1	0.68	0.44	0.54	142
Avg/total	0.71	0.72	0.70	391

Accuracy: 72.12%

Table 5 – SVM with tuned parameters: C=100, gamma = 0.1, kernel = 'rbf'

	Precision	Recall	F1-score	Support
0	0.81	0.95	0.88	252
1	0.87	0.60	0.71	139
Avg/total	0.83	0.83	0.82	391

Accuracy: 82.61%

After tuning parameters, SVC behaviour is far more better than in previous case. Logistic Regression metrics show us that the behaviour didn't change too much according to the previous case.

To sum up, the best performance has Support Vector Classifier. Acording to dataset, I'll rather use SVM than Logistic Regression classfier because number of features is small and the number of training examples is intermediate in my opinion.

RESPONSE TO THE 10 NEW DISCOVERED PASSENGERS

Table 6 – Logistic Regression - output

Actual output	Model output	Probability 0	Probability 1
0	0	0.74311171	0.25688829
0	0	0.74311171	0.25688829
1	0	0.52780567	0.47219433
0	0	0.72979703	0.27020297
0	0	0.75598966	0.24401034
0	0	0.75598966	0.24401034
1	1	0.40833927	0.59166073
1	0	0.55850184	0.44149816
1	1	0.42501415	0.57498585
1	1	0.26011594	0.73988406

Table 7 – SVC - output

Actual output	Model output	Probability 0	Probability 1
0	0	0.98511669	0.01488331
0	0	0.98511669	0.01488331
1	0	0.78656724	0.21343276
0	0	0.96557864	0.03442136
0	0	0.99201528	0.00798472
0	0	0.99201528	0.00798472
1	1	0.19527331	0.80472669
1	0	0.78665345	0.21334655
1	1	0.15862112	0.84137888
1	1	0.21237841	0.78762159

Naive Bayes algorithm doesn't behave good when asked to predict probabilites, so the outputs aren't taken too seriously.

CONCLUSION

The task was to create the system which would predict probability of passenger survival. It's important to have clean dataset, so preprocessing is one of the most important things when implementing machine learning algorithms. Acording to given dataset, the particular algorithms had been choosen and implemented. Because Naive Bayes algorithm doesn't perform well for predicting probabilities, that algorithm is not considered anymore. The other two algorithms perform similarly to each other (if parameters aren't tuned), but SVC has better F1 score. Futhermore, after parameter tuning, SVC has perform far more better than Logistic Regression model. Generally, we rather use SVC than Logistic Regression when we have dataset like this: small number of features and intermediate number of training examples.