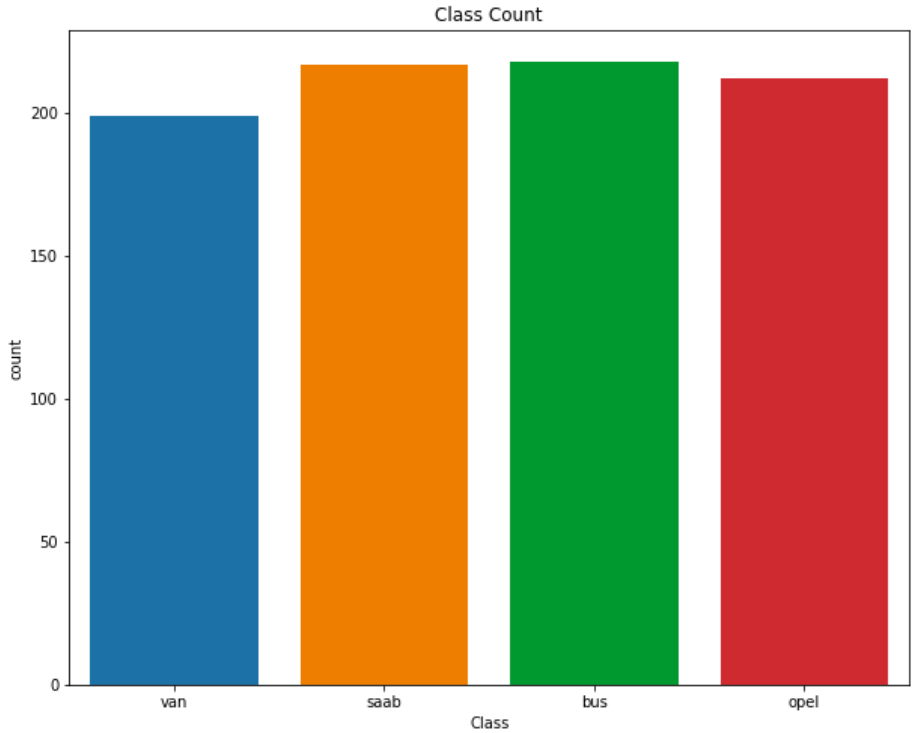
Introduzione

Il dataset contiene le caratteristiche dei veicoli estratte dalle sagome delle immagini dei veicoli dall'estensione HIPS (Hierarchical Image Processing System). Tale dataset riporta sia le misure classiche basate sui momenti, come la varianza e la curtosi degli assi maggiori/minori, sia le misure euristiche come cavità, circolarità e compattezza.
L'obiettivo è classificare i veicoli sulla base delle loro caratteristiche.

Analisi Descrittiva

Di seguito sono riportati il numero di veicoli per ogni categoria di veicoli



Si hanno quindi quattro tipologie di veicoli distinti, ossia i bus, Saab 9000, Opel manta 4000 e Cheverolet van.

Prepocessing

Denotiamo con y il vettore delle risposte, ossia le tipologie di veicoli, e denotiamo con X la matrice delle caratteristiche ricavate. Successivamente, viene applicata la standardizzazione dei dati. Per ultimo, i dati vengono divisi in dati di training e dati di test

```
y = data["Class"]
X = data[['COMPACTNESS','CIRCULARITY','DISTANCE_CIRCULARITY','RADIUS_RATIO',"PR.AXIS_ASPECT_RATIO","MAX.LENGTH_ASPECT_RATIO","SCATTER_RATIO","
```

```
# initialising the MinMaxScaler
scaler = MinMaxScaler(feature_range=(0,1))
X = scaler.fit_transform(X)
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=42)
```

Regressione Logistica

La regressione logistica è una delle tecniche più semplici e intuitive di classificazione. L'idea di base consiste nello stimare le probabilità che una certà unità appartenga ad una data classe e si definisce la regola di decisione, ossia l'unità verrà assegnata alla classe di probabilità maggiore. Tradotto in temini matematici si ha che

$$d(x) = argmax_{j=1,\ldots,K} P(Y = j|x)$$

Nel caso della regressione logistica multinomiale si assume che la probabilità possa essere stimata con la funzione softmax

Regressione Logistica

Si carica il modello di regressione logistica e lo si allena sui dati di allenamento

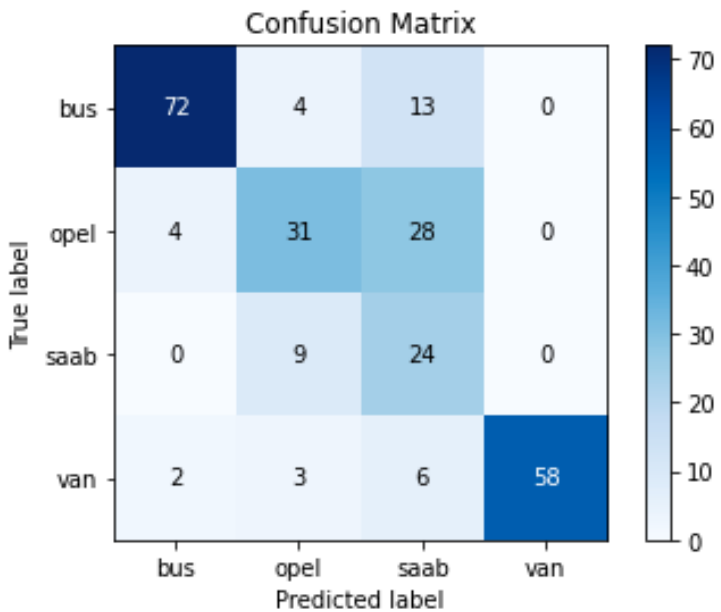
```
# Modello
logistic = LogisticRegression(multi_class='multinomial', solver='lbfgs')
# Allenamento sui dati
logistic.fit(X_train,y_train)
# Previsione sui dati di test
p_test=logistic.predict(X_test)
```

Regressione Logistica

Vengono riportat i gli indicatori principali della regressione logistica

	precision	recall	f1-score	support
bus	0.92	0.81	0.86	89
opel	0.66	0.49	0.56	63
saab	0.34	0.73	0.46	33
van	1.00	0.84	0.91	69
accuracy			0.73	254
macro avg	0.73	0.72	0.70	254
weighted avg	0.80	0.73	0.75	254

e la matrice di confusione



K-Nearest Neighbors

In metodo K-Nearest Neighbors (abbreviato KNN) è una tecnica che, come la regressione logistica, stima le probabilità che una data unità appartenga ad una certa classe. L'idea alla base è che ,dato un certo valore intero non negativo K, KNN considera i K punti più vicini all'unità che vogliamo classificare, rispetto ad una certa distanza di solito quella euclidea, e approssima la probabilità che l'unità appartenga ad una data classe j con la percentuale di unità che appartengono alla classe scelta.

K-Nearest Neighbors

K-Nearest Neighbors

```
parameters = {
     "n_neighbors": range(1, 100),
     "weights": ["uniform", "distance"],
 }
knn = GridSearchCV(KNeighborsClassifier(), parameters)
knn.fit(X_train, y_train)
print(knn.best_params_)
p_test = knn.predict(X_test)
```

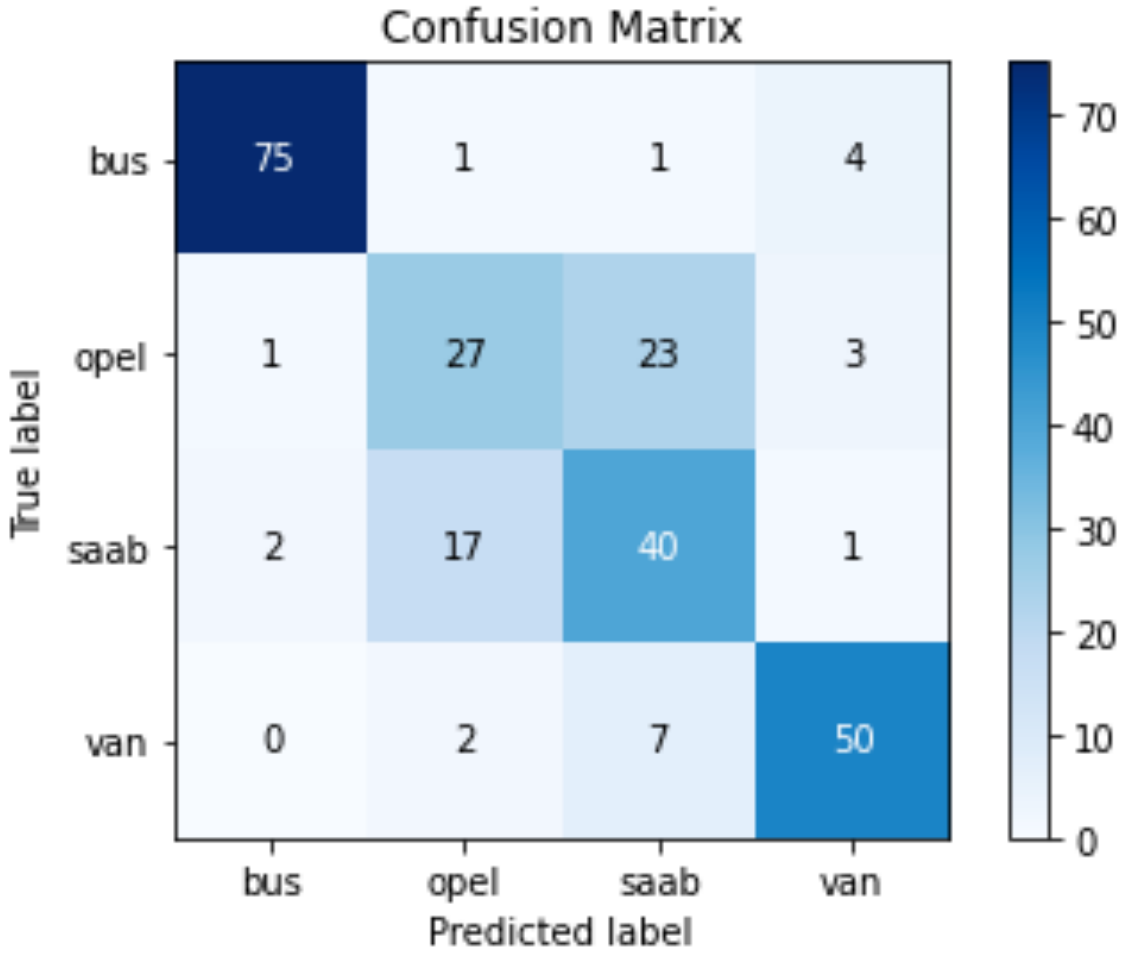
```
{'n_neighbors': 3, 'weights': 'distance'}
```

```
report=classification_report(p_test, y_test)
print(report)
```

```
              precision    recall  f1-score   support

         0.0       0.96      0.93      0.94        81
         1.0       0.57      0.50      0.53        54
         2.0       0.56      0.67      0.61        60
         3.0       0.86      0.85      0.85        59

    accuracy                           0.76       254
   macro avg       0.74      0.74      0.74       254
weighted avg       0.76      0.76      0.76       254
```

K-Nearest Neighbors

Matrice di confusione per KNN



Alberi di decisione

Gli alberi di decisione, in particolar modo quelli di classificazione, suddividono lo spazio in delle regioni (di solito di forma rettangolare) e, a seconda della regione in cui cade una certa unità, viene associata ad essa la classe più frequente di quella regione. Equivalentemente, l'obiettivo è minimizzare degli indici che misurano "l'impurità" di un nodo; tali indici sono o l'indice di Gini

$$G = \sum_{j=1,\dots,K} p_j(1 - p_j)$$

che misura la varianza totale del nodo, oppure l'entropia

$$H = -\sum_{j=1,\dots,K} p_j \log(p_j)$$

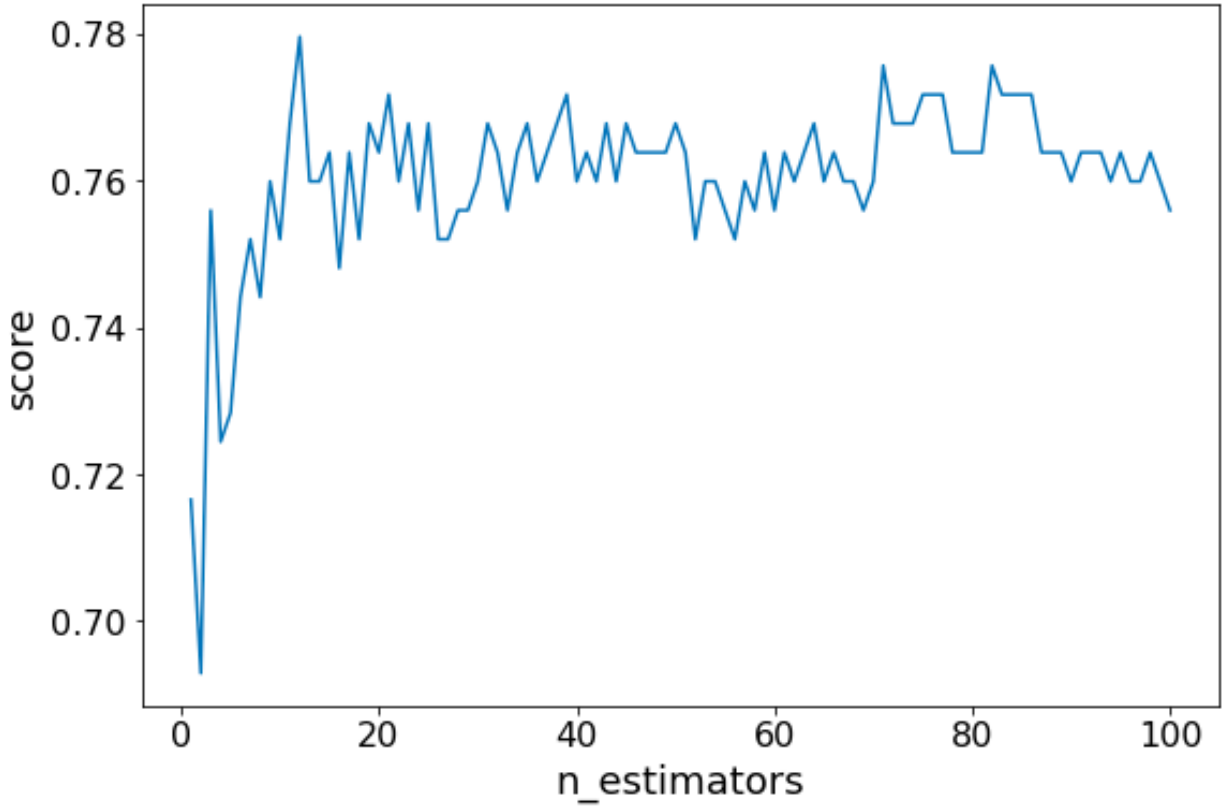
che tende invece a penalizzare probabilità basse.

Bagging

Il bagging è una generalizzazione degli alberi di decisione. Tale metodo allena in parallelo più alberi di decisione e poi li aggrega prendendone il valore più frequente. Questa procedura causa un decremento della varianza.

Bagging

Dal grafico

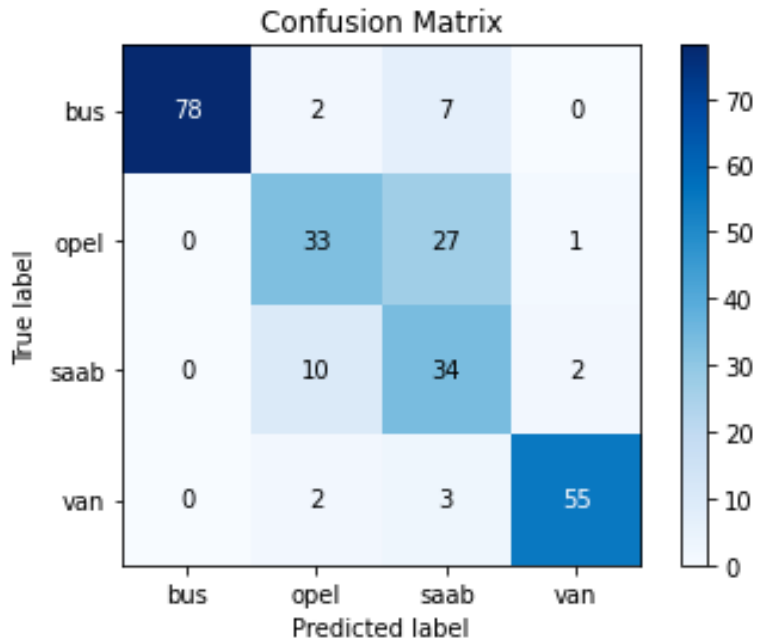


si vede che il numero ottimale di stimatori è 18

Bagging

Vengono riportati i valori degli indicatori e la matrice di confusione

	precision	recall	f1-score	support
bus	0.90	1.00	0.95	78
opel	0.54	0.70	0.61	47
saab	0.74	0.48	0.58	71
van	0.92	0.95	0.93	58
accuracy			0.79	254
macro avg	0.77	0.78	0.77	254
weighted avg	0.79	0.79	0.78	254



Support Vector Machine

Le support vector machine, abbreviate SVM, permettono di classificare le unità sulla base della loro posizione rispetto ad un iperpiano; nel caso binario, infatti, l'unità avrà classe positiva se si trova al di sopra dell'iperpiano e avrà classe negativa altrimenti. Se abbiamo più di due classi, come nel caso di studio che stiamo analizzando, si considera una versione generalizzata.

Support Vector Machine

Anzitutto viene caricato il modello, viene allenato sui dati di allenamento e vengono stampati a schermo i valori ottimali

```
params_grid = {'C': [0.1,1, 10, 100], 'gamma': [1,0.1,0.01,0.001],'kernel': ['rbf', 'poly', 'sigmoid',"linear"]}
svm = GridSearchCV(SVC(), params_grid, cv=5)
svm.fit(X_train,y_train)
p_test=svm.predict(X_test)
```

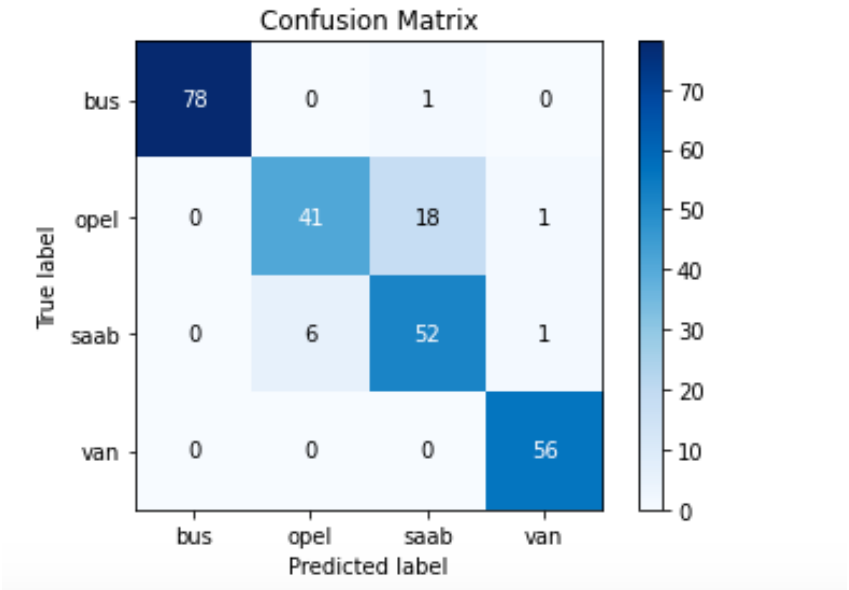
```
# View the accuracy score
print('Best score for training data:', svm.best_score_,"\n")

# View the best parameters for the model found using grid search
print('Best C:',svm.best_estimator_.C,"\n")
print('Best Kernel:',svm.best_estimator_.kernel,"\n")
print('Best Gamma:',svm.best_estimator_.gamma,"\n")
```

```
Best score for training data: 0.8158809286426434

Best C: 100

Best Kernel: rbf

Best Gamma: 1
```

Support Vector Machine

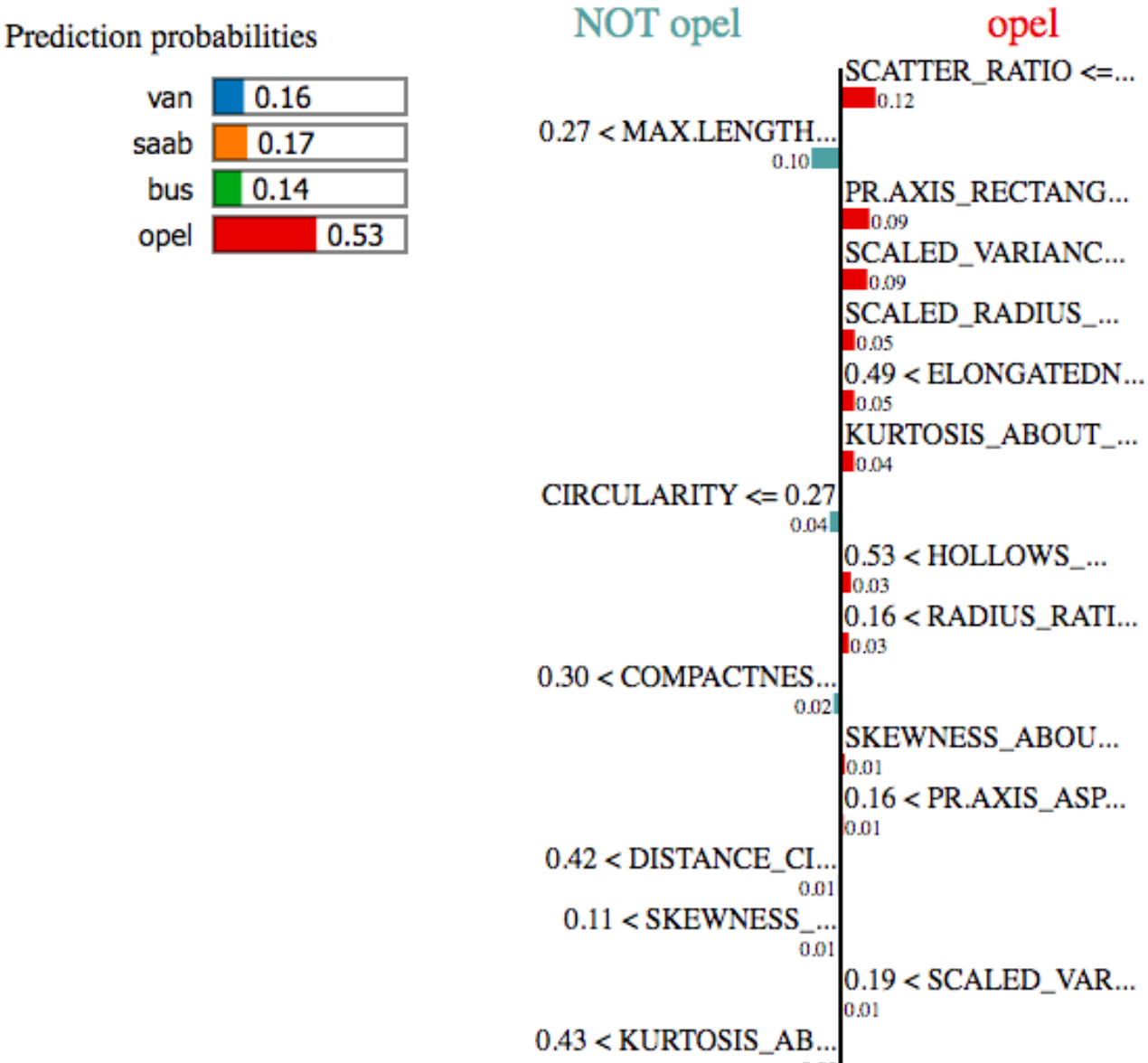
Vengono riportati gli indicatori e la matrice di confusione per le SVM

	precision	recall	f1-score	support
bus	1.00	0.99	0.99	79
opel	0.87	0.68	0.77	60
saab	0.73	0.88	0.80	59
van	0.97	1.00	0.98	56
accuracy			0.89	254
macro avg	0.89	0.89	0.89	254
weighted avg	0.90	0.89	0.89	254



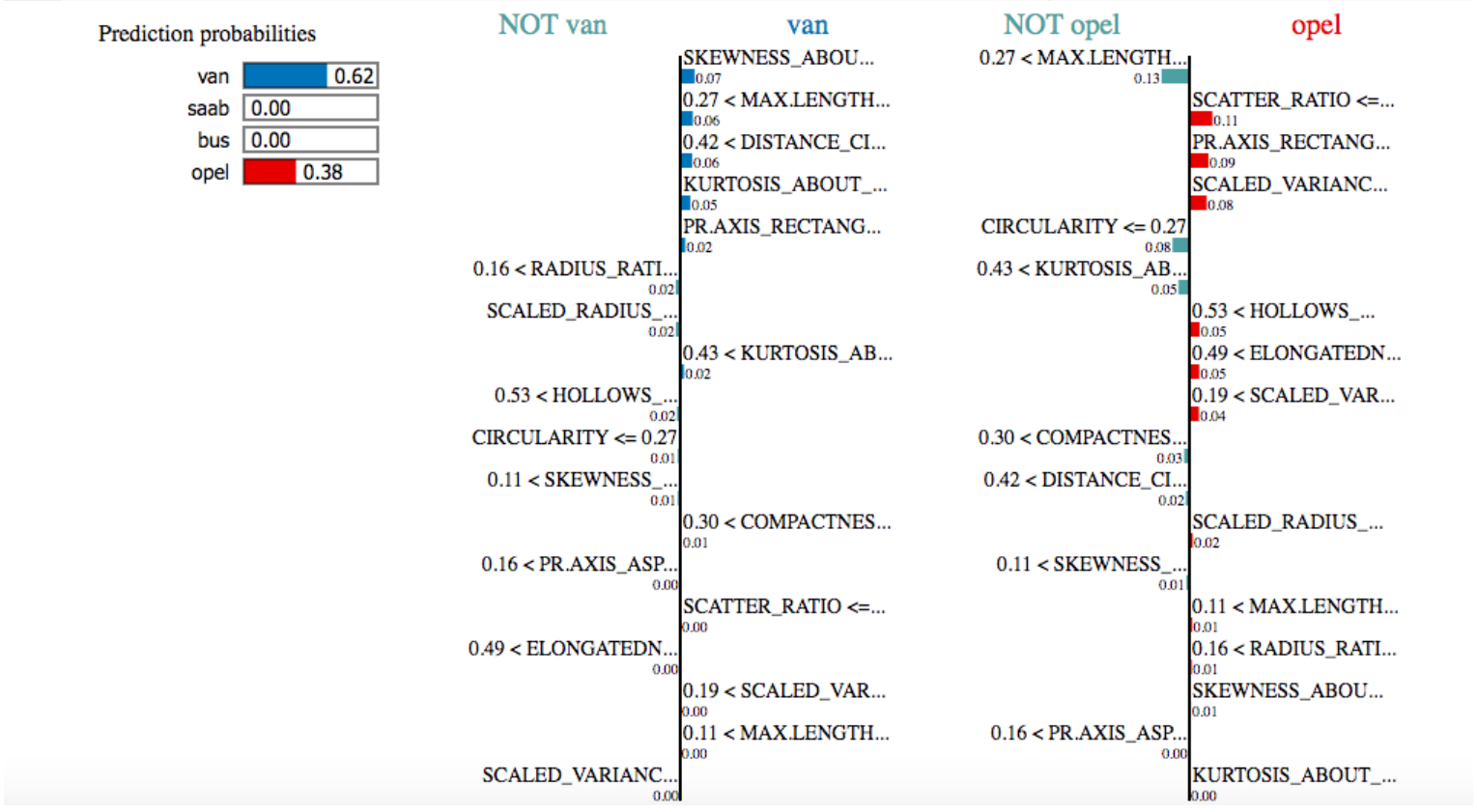
Interpretabilità Regressione logistica

Viene applicato il metodo LIME per determinare l'importanza di una variabile sulla stima delle probabilità



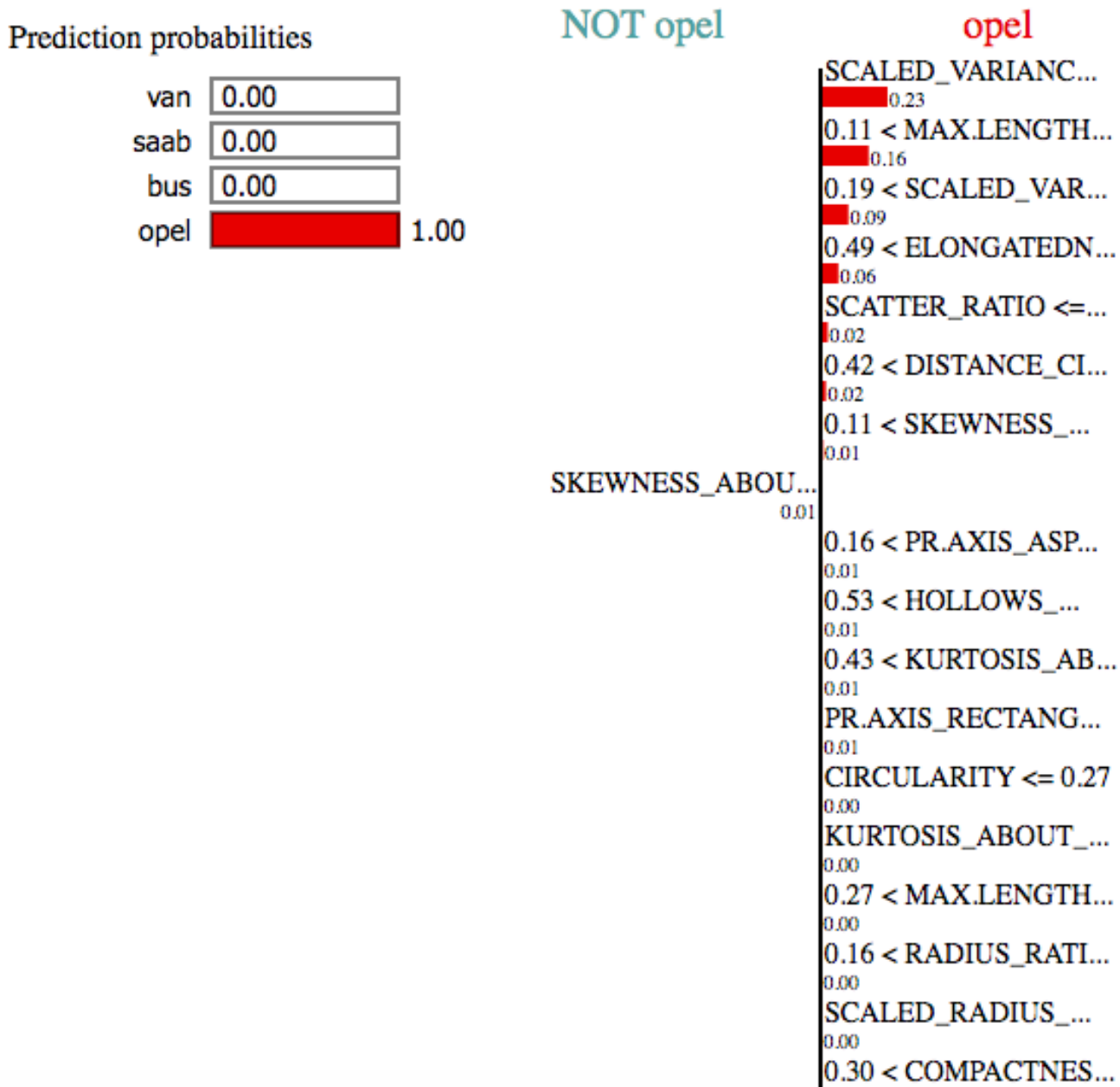
Interpretabilità KNN

Matrice di confusione per KNN

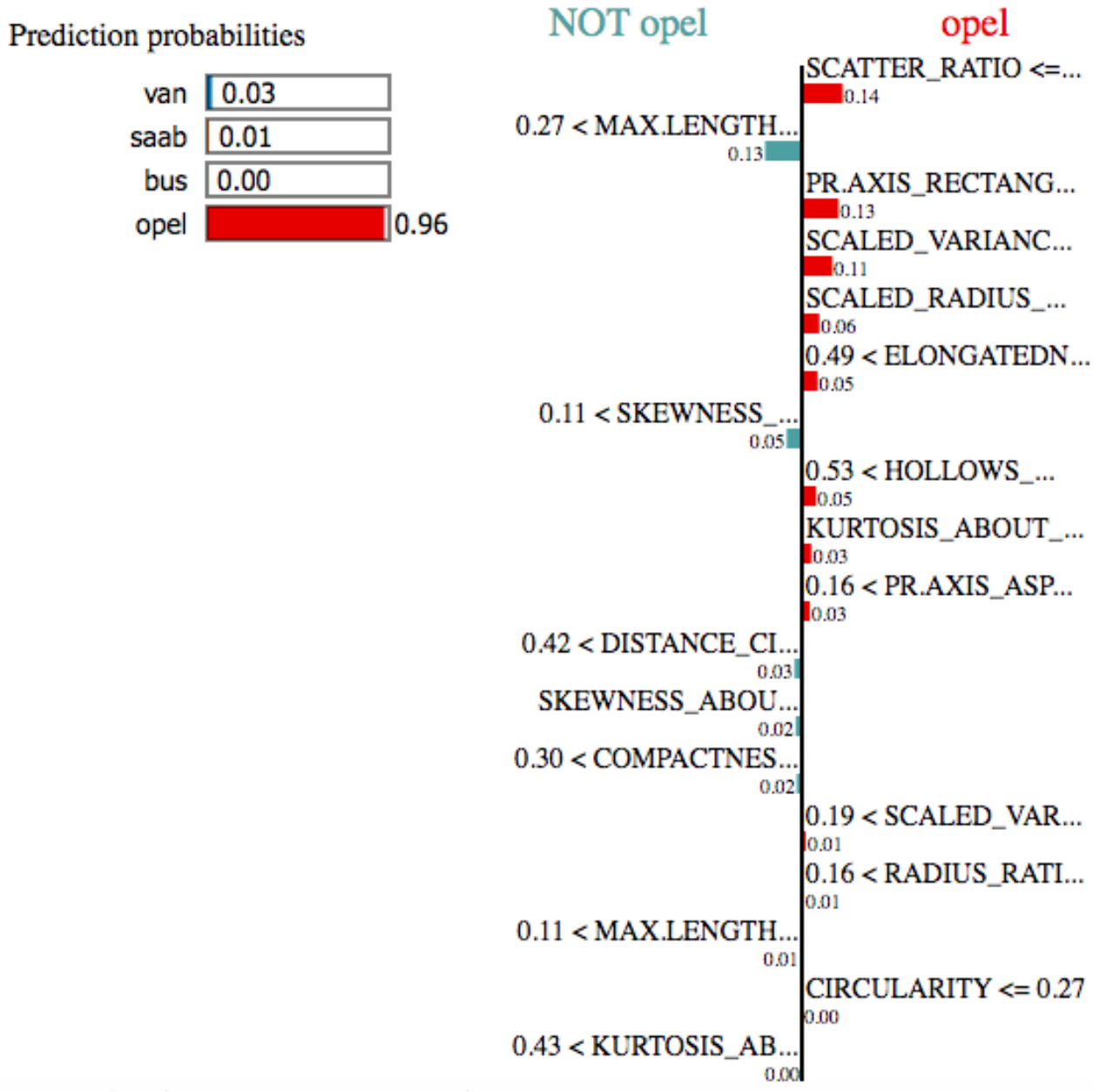


Interpretabilità Bagging

Vengono riportati i valori degli indicatori e la matrice di confusione



Interpretabilità SVM



Risultati finali

Per ultimo, vengono riportate le accuratezze di ogni modello.

	Model	Accuracy
1	Logistic	0.728346
2	KNN	0.755906
2	SVM	0.893701
2	Bagging	0.775591

La SVM è il modello con la maggiore accuratezza, pari a circa 0.9