Reconocimiento Óptico de Dígitos escrito a mano

Andrea Vargas Gutiérrez 132192

Introducción.

En este projecto se utilizarán varias técnicas de Aprendizaje de Máquina para reconocer distintos dígitos escritos a mano. La base con la que se trabaja fue preprocesada por programas hechos por NIST para extraer mapas de bits normalizados de digitos escritos a mano.
Para la formación de la base de datos, participaron 43 personas distintas, de las cuales 30 contribuyen al set de entrenamiento del modelo y 13 distintas al set de prueba. Esto de tal forma que no entrenemos el modelo con los mismos tipos de escritura de cierta persona.

Datos

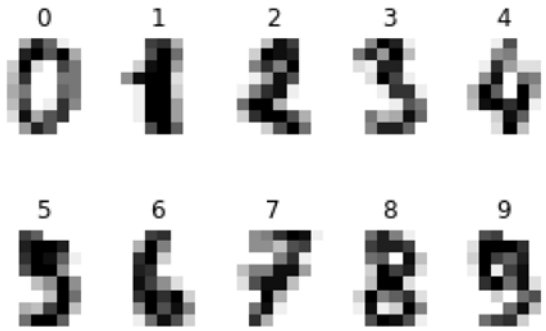
Los datos se encontraban previamente divididos en Test y Train. Fueron descargados de https://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/ (https://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/) Se utilizó el archivo optdigits.tes para los datos de prueba y optdigits.tra para el set de entrenamiento. (3823 en training set y 1797 datos en test set)

</p> Para la representación gráfica de los números se utilizó el set de datos digits proporcionado por sklearn, el cual coincide con los datos descargados para el set de prueba.

Forma en que se presentan los datos:

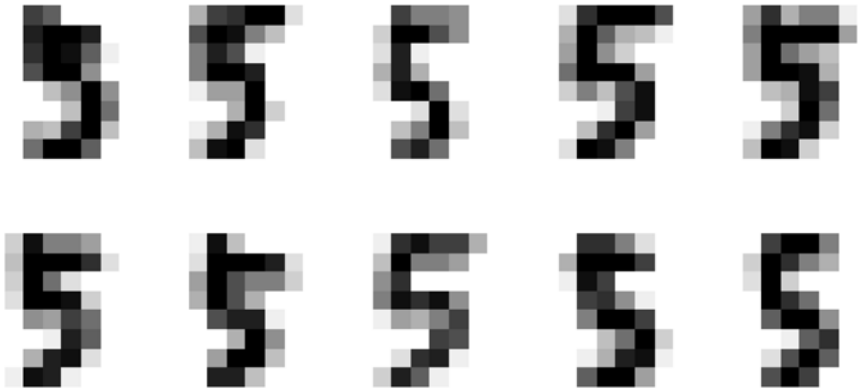
Los mapas de bits se normalizaron a un tamaño de 32x32. Estos se dividieron en bloques disjuntos de 4x4. Esto da como resultado una matriz de 8x8 donde cada elemento, contiene el número de pixeles activos para cada bloque, es decir un número entre el 0 y 16.

Las representaciones gráficas de los números del 0 al 9 se ven de la siguiente manera.



Metodología

Los números escritos a mano pertenecientes a la misma clasificación pueden tener formas muy distintas entre sí. Para ver ésto más claro, se presentan gráficamente los primeros 10 números "5".



La metodología que usaremos, pretende reconocer las características generales para cada una de las etiquetas (números) de nuestros datos y así predecir a qué numero corresponde una nueva representación gráfica. Para hacer esto utilizaremos 3 modelos: Bosques aleatorios, K vecinos cercanos y Maquinas de soporte vectorial.

Estimación de Hiperparámetros

Para determinar los hiperparámetros, se dividieron los datos en 5 bloques y se utilizó VALIDACION CRUZADA para determinar la precisión del modelo. Se optó por éste método de estimación porque realiza una prueba sobre datos "no vistos" y promedia sobre varias pruebas con los datos. Con esto se espera que los hiperparámetros elegidos sirvan para describir mejor los nuevos datos.

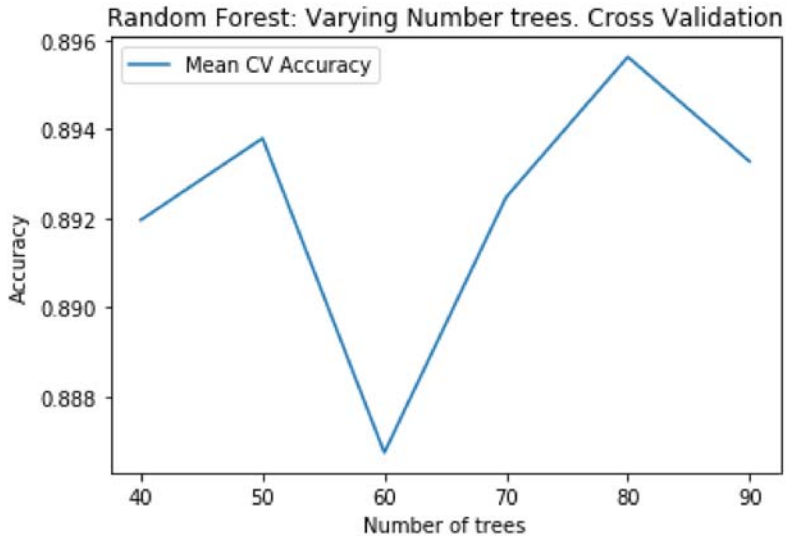
Para el modelo de Bosques aleatorios se establecio como hiperparámetro el número de árboles en el bosque, probando con un rango de 40 a 90 arboles con saltos de tamaño 10. Se obtuvo que el parámetro que tiene una mayor presición media es con 60 arboles. Más adelante se presentarán las gráficas con las distintas exactitudes promedio que se utilizaron para la elección de los hiperparámetros óptimos.

Para el caso de Knn-neighbors el hiperparámetro del modelo fue el número de vecinos que se tomaría en cuenta para la clasificación de la nueva observación. Se utilizó un rango de 1 a 15 vecinos y se obtuvo una precisión máxima en 1 vecino cercano. Después presenta otro pico en 3 vecinos cercanos y luego, como es de esperarse, tiene una tendencia descendente con más vecinos ya que el modelo se sobreajusta.

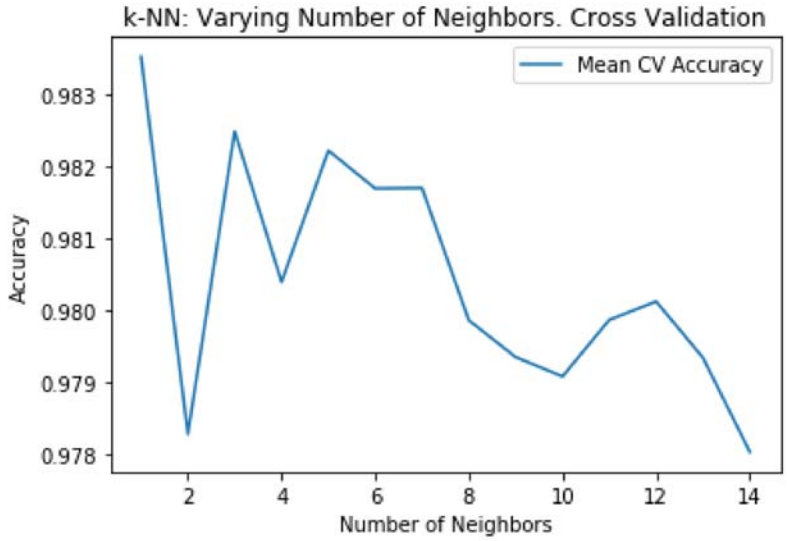
En el modelado de SVM se utilizó como hiperparámetro el tipo de Kernel del modelo. Se propusieron las siguientes opciones: 'linear', 'rbf' , 'sigmoid' y 'Polinomial'. Para el tipo de Kernel Polinomial se probó ajustando distintos grados del polinomio (del 2 al 6).

Graficas de hiperparámetros

Random Forest

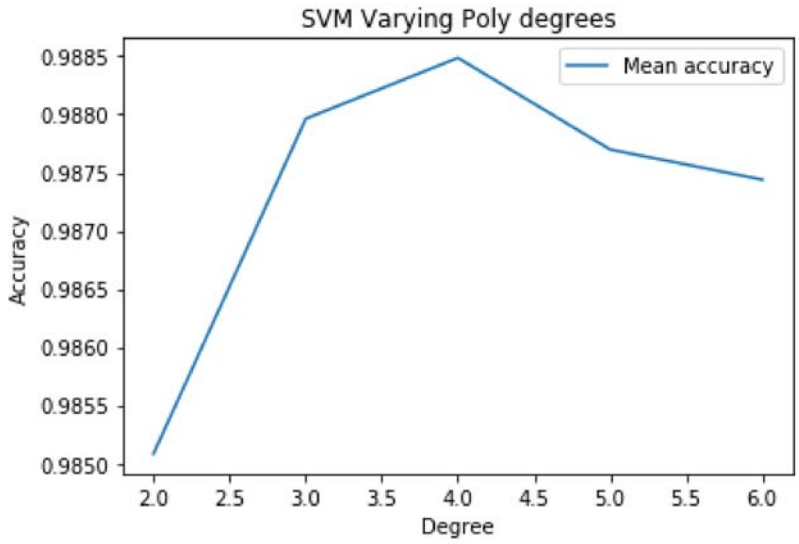


Knn



SVM

```
linear mean accuracy with 5 folds: 0.978021822526
rbf mean accuracy with 5 folds: 0.665490158453
sigmoid mean accuracy with 5 folds: 0.10175281863
Poly degree= 4 mean accuracy with 5 folds: 0.988486309644
```



Una vez seleccionados los hiperparámetros, se entrenan los distintos modelos con todos los datos de entrenamiento. Ahora es momento de probar qué tan buenos fueron nuestros modelos para predecir nuevas observaciones nunca antes vistas.

Resultados

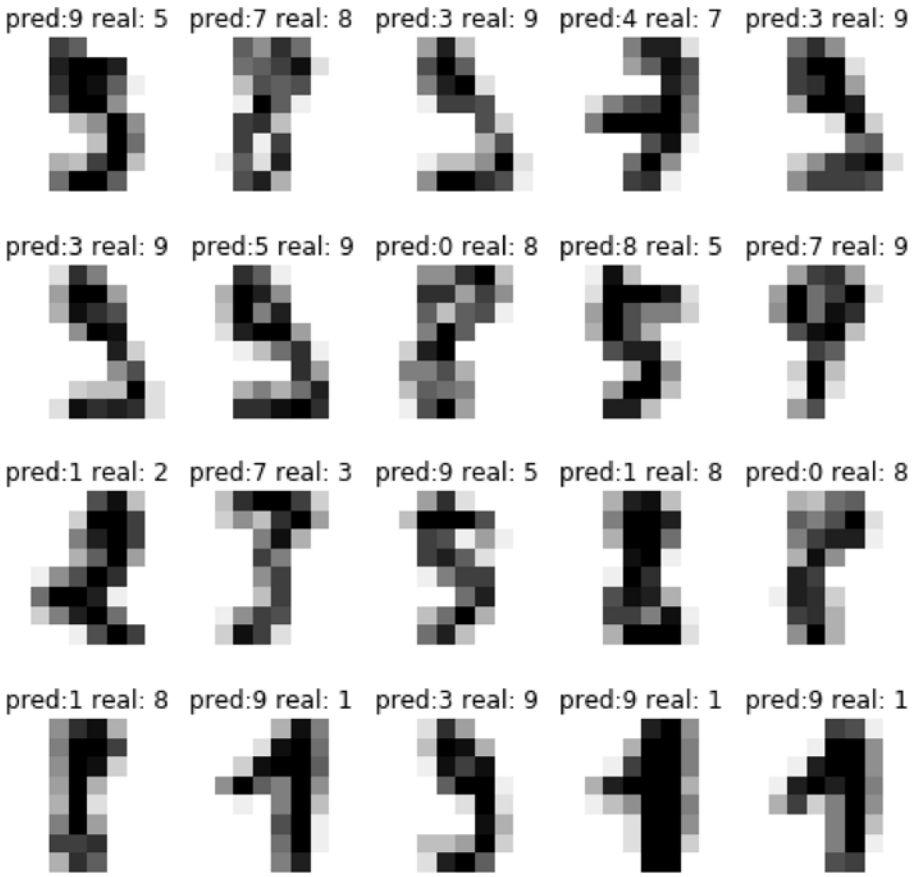
RANDOM FOREST

El modelo de bosque aleatorio presentó una precisión de predicción del 88.7% lo cual representa un buen desempeño. Se presenta la matriz de confusión y un reporte de indicadores para ver el desempeño del modelo. Puede verse como los números 9 y 1 tuvieron mayor indice de error, confundiendose principalmente con el número 3 y 9 respectivamente

```
Random Forest accuracy: 0.889259877574

[[176   0   0   0   2   0   0   0   0   0]
 [  0 139  15   3   1   1  10   2   0  11]
 [  1   2 164   4   0   0   2   3   0   1]
 [  1   2   3 164   0   3   0   5   2   3]
 [  0   1   0   0 166   0   2  12   0   0]
 [  4   0   0   0   1 156   2   1   2  16]
 [  2   1   0   0   1   0 176   0   1   0]
 [  0   0   0   0   6   1   0 166   1   5]
 [  2   8   8   3   3   1   0   6 141   2]
 [  1   1   0  16   1   1   0   9   1 150]]
             precision    recall  f1-score   support

          0       0.94      0.99      0.96       178
          1       0.90      0.76      0.83       182
          2       0.86      0.93      0.89       177
          3       0.86      0.90      0.88       183
          4       0.92      0.92      0.92       181
          5       0.96      0.86      0.90       182
          6       0.92      0.97      0.94       181
          7       0.81      0.93      0.87       179
          8       0.95      0.81      0.88       174
          9       0.80      0.83      0.82       180

avg / total       0.89      0.89      0.89      1797
```

De las 199 predicciones incorrectas, se presentan las representaciones gráficas de las primeras 20 para juzgar personalmente qué tan legible era realmente el número presentado. Con estas representaciones podemos ver como por ejemplo, los 9 y los 5 realmente parecen tener una relación gráfica muy similar.

```
199 predicciones incorrectas
Primeras 20 predicciones incorrectas
```

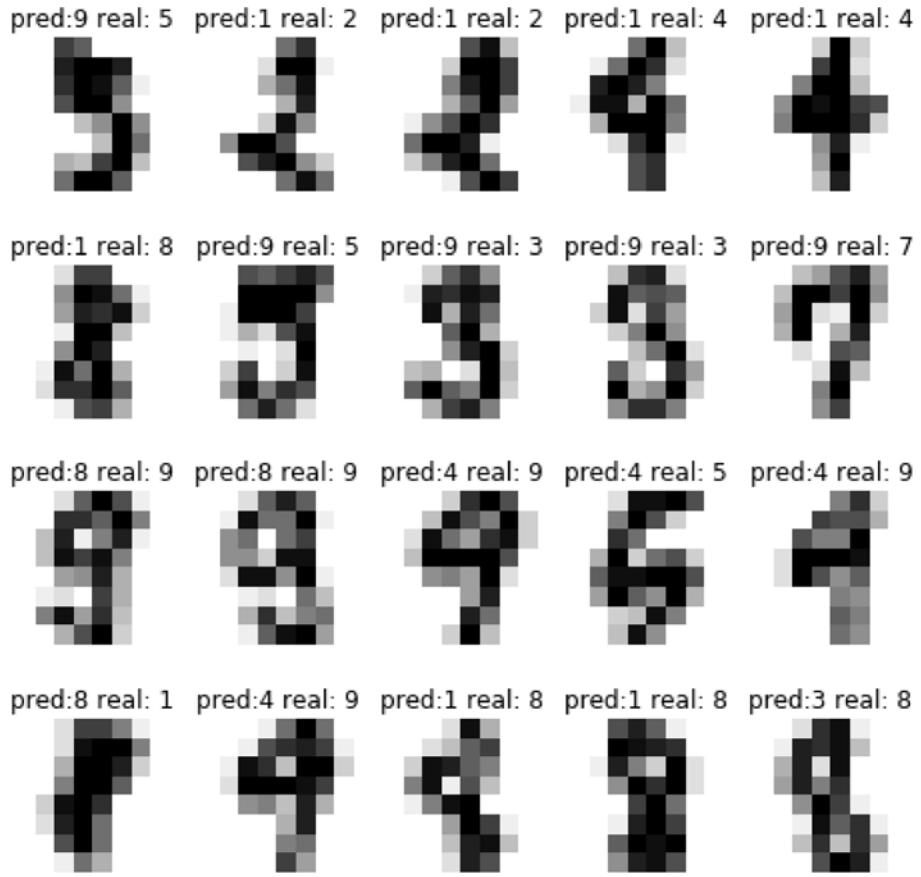


Knn

Usando el método de K vecinos cercanos podemos ver cómo se incrementa la precisión del modelo en comparación del bosque aleatorio. Para este modelo la precisión es de 97% presentando únicamente36 diagnosticos incorrectos de 1797 nuevas observaciones. El número más confundido con éste modelo es nuevamente el 9 y el 8

```
Knn accuracy: 0.979966611018
[[178   0   0   0   0   0   0   0   0   0]
 [  0 181   0   0   0   0   0   0   1   0]
 [  0   2 175   0   0   0   0   0   0   0]
 [  0   0   0 179   0   0   0   2   0   2]
 [  0   2   0   0 178   0   0   0   1   0]
 [  0   0   0   0   1 179   0   0   0   2]
 [  0   0   0   0   0   0 181   0   0   0]
 [  0   0   0   0   0   0   0 177   0   2]
 [  0   8   0   1   0   0   0   0 164   1]
 [  0   0   0   3   3   2   0   0   3 169]]
             precision    recall  f1-score   support

          0       1.00      1.00      1.00       178
          1       0.94      0.99      0.97       182
          2       1.00      0.99      0.99       177
          3       0.98      0.98      0.98       183
          4       0.98      0.98      0.98       181
          5       0.99      0.98      0.99       182
          6       1.00      1.00      1.00       181
          7       0.99      0.99      0.99       179
          8       0.97      0.94      0.96       174
          9       0.96      0.94      0.95       180

avg / total       0.98      0.98      0.98      1797
```

```
36 predicciones incorrectas
A continuación se presentan los primeros 20 errores del modelo
```



SVM

Ahora entrenaremos el modelo utilizando el metodo SVM. Para esto se eligió un Kernel polinomial de grado 4.

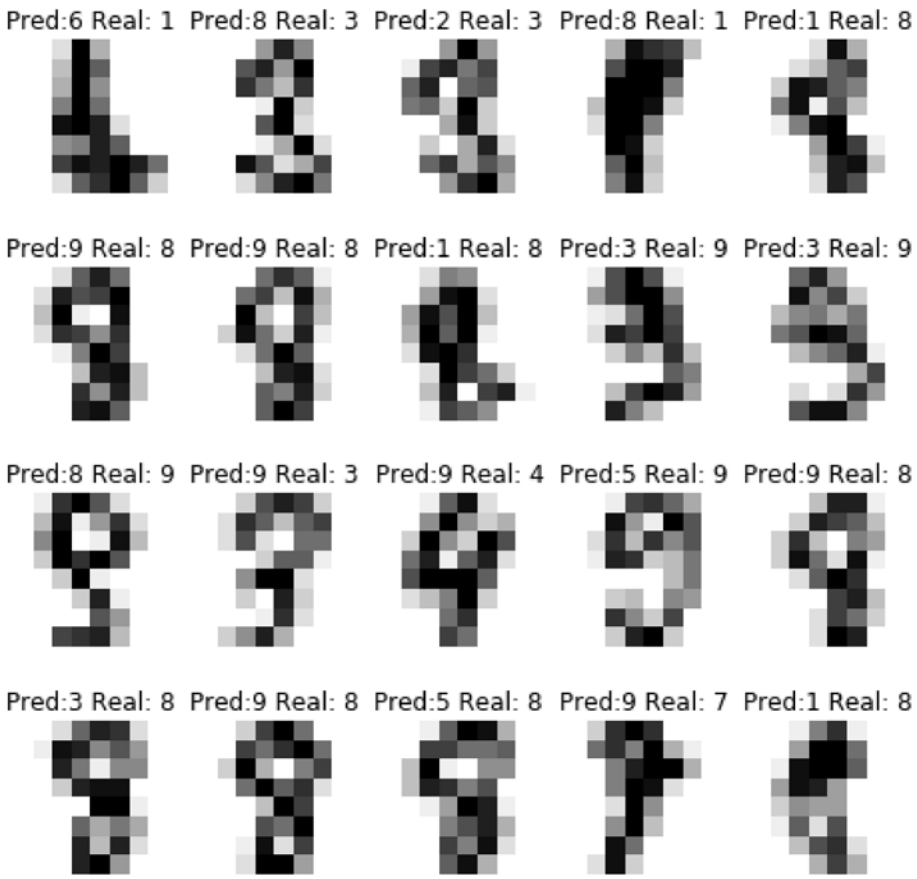
La precisión de este modelo es poco menor a la de k vecinos sin embargo sigue siendo muy alta 97% . Para este set de datos se clasificaron mal 44 números. Siendo los más confundido el 7,8 y nuevamente el 9

```
test accuracy: 0.9755147468
```

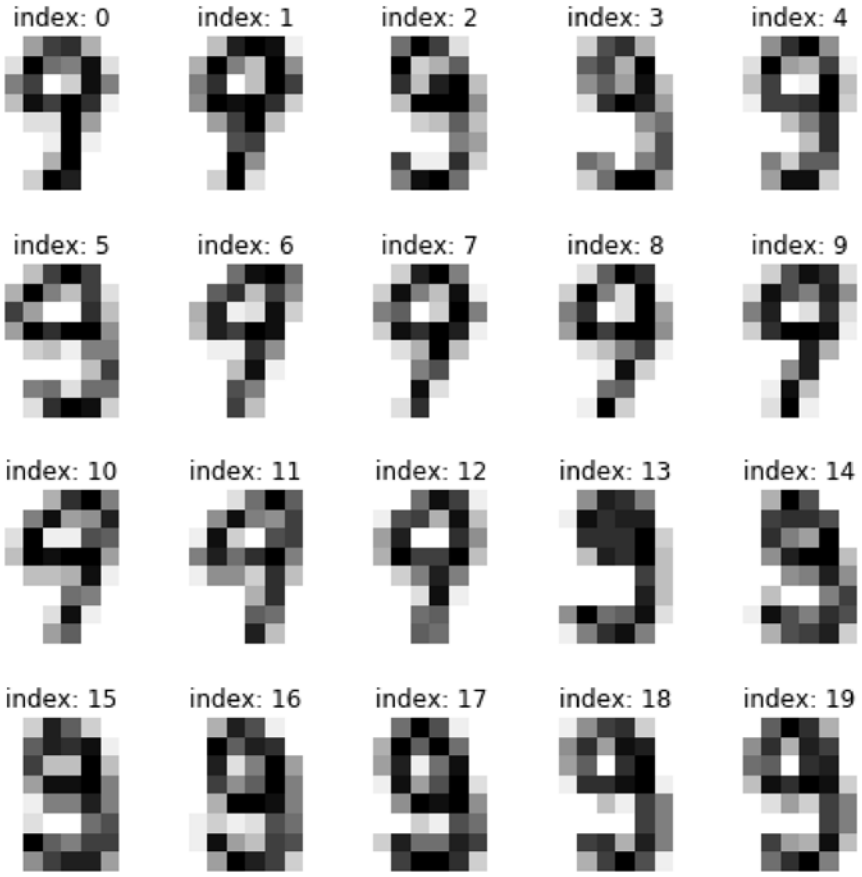
```
array([[178,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0, 180,   0,   0,   0,   0,   1,   0,   1,   0],
       [  0,   1, 176,   0,   0,   0,   0,   0,   0,   0],
       [  0,   0,   1, 176,   0,   1,   0,   1,   2,   2],
       [  0,   0,   0,   0, 179,   0,   0,   0,   1,   1],
       [  1,   0,   0,   0,   0, 180,   0,   0,   0,   1],
       [  0,   0,   0,   0,   0,   0, 181,   0,   0,   0],
       [  0,   0,   0,   0,   0,   4,   0, 169,   0,   6],
       [  0,   6,   0,   1,   0,   1,   0,   0, 162,   4],
       [  0,   0,   0,   4,   0,   3,   0,   0,   1, 172]], dtype=int64)
```

	precision	recall	f1-score	support
0	0.99	1.00	1.00	178
1	0.96	0.99	0.98	182
2	0.99	0.99	0.99	177
3	0.97	0.96	0.97	183
4	1.00	0.99	0.99	181
5	0.95	0.99	0.97	182
6	0.99	1.00	1.00	181
7	0.99	0.94	0.97	179
8	0.97	0.93	0.95	174
9	0.92	0.96	0.94	180
avg / total	0.98	0.98	0.98	1797

44 predicciones mal
Primeras 20 predicciones incorrectas



Dado lo observado en los 3 modelos, se decidió hacer una muestra de números 9 para poder ver por qué presentan tanto error en la clasificación



Después de ver las distintas representaciones del número 9 podemos entender por qué resulta el número más confundido. Por ejemplo, en el 6to número podría confundirse con un 4 o el ejemplo 17 o 14 podrían ser un 8.

Conclusion

Los modelos presentados tuvieron una alta tasa de predicciones correctas. Esto es debido a que los elementos que se modelan son bastante uniformes y estandarizados. Es decir, no tienen mucho ruido y no se presentan sobre ajustes ni bajo ajustes por la selección de las variables en los modelos. Esto también debido a que no hay variables (columnas) linealmente dependientes de otras.

Si bien es un ejercicio simple en cuanto a elección de variables para tomar en cuenta en el modelo, tiene detrás un gran trabajo en la estandarización de los datos y modelado de las representaciones gráficas como arreglos para poder ser utilizados en los modelos predictivos.

La elección de los hiperparámetros también juega un papel importante para uno correcto entrenamiento del modelo. Esto se puede ver principalmente en el método de SVM en el que se puede llegar a tener un 97.5% de precisión o un 60% o 10% si se elige un mal Kernel

Referencias

- Presentaciones del profesor Fernando Esponda para el curso Aprendizaje de Máquina, Otoño 2017 , ITAM
- https://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/ (https://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/)